GENERALIZABILITY OF STATISTICAL PREDICTION FROM PSYCHOLOGICAL
ASSESSMENT DATA: AN INVESTIGATION WITH THE MMPI-2-RF

A dissertation submitted
to Kent State University in partial
fulfillment of the requirements for the
degree of Doctor of Philosophy

by

William H. Menton

August 2019

© Copyright

All rights reserved

Except for previously published materials

Dissertation written by

William H. Menton

B.A., University of Minnesota, 2005

M.A., Kent State University, 2016

Ph.D., Kent State University, 2019

Approved by

Yossef Ben-Porath, Chair, Doctoral Dissertation Committee

Mary Beth Spitznagel, Members, Doctoral Dissertation Committee

Manfred van Dulmen

Richard Serpe

Michael Ensley

Accepted by

Maria Zaragosa, Chair, Department of Psychological Sciences

James Blank, Dean, College of Arts and Sciences

TABLE OF CONTENTS	iii
LIST OF TABLES	iv
ACKNOWLEDGEMENTS	vi
I. INTRODUCTION	1
II. METHOD	22
III. RESULTS	50
IV. DISCUSSION	82
V. REFERENCES	101
VI. APPENDIX: R Code	110

LIST OF TABLES

Table 1. Sample Exclusion Rates and Demographic Characteristics	22
Table 2. Prediction Criteria and Their Compositions	25
Table 3. Distributional Characteristics of Prediction Criteria	26
Table 4. Modeling Techniques Used in Present Study	29
Table 5. Zero-Order Correlations Between RRF Scales and MMPI-2-RF Substantive Scales ..	41
Table 6. AUCs Between Diagnostic Groups and MMPI-2-RF Substantive Scales	43
Table 7. Single-Predictor Models	45
Table 8. Manually Selected Predictor Variables for Theory-Driven Models	47
Table 9. Regression Models Predicting RRF Depression	53
Table 10. Regression Models Predicting RRF Substance Abuse	55
Table 11. Regression Models Predicting RRF Delusions	57
Table 12. Regression Models Predicting RRF Pain	59
Table 13. External Validation (VA Medical Center) Performance of Regression Models	61
Table 14. Rank Sum Scores for External Validation Performance of Regression Models Based on RMSE	65
Table 15. Classification Models Predicting Depressive Disorder Diagnosis	69
Table 16. Classification Models Predicting Substance Use Disorder Diagnosis	71
Table 17. Classification Models Predicting Thought Disorder Diagnosis	73
Table 18. Classification Models Predicting Somatic Disorder Diagnosis	75
Table 19. External Validation Performance of Classification Models (Overall Classification Accuracy)	77

Table 20. External Validation Performance of Classification Models (Area Under the Curve)	78
Table 21. Rank Sum Scores for External Validation Performance of Classification Models Based on AUC	81
Table 22. External Validation RMSE Of Random Forests Regression for Different Numbers of Trees	94

Acknowledgements

I would like to thank my advisor, Dr. Yossef Ben-Porath, whose guidance, support, and mentorship made this project possible. I am also grateful for the support and feedback provided by my dissertation committee: Dr. Manfred van Dulmen, Dr. Mary Beth Spitznagel, and Dr. Richard Serpe. Finally, I wish to gratefully acknowledge my former mentor, the late Dr. Will Grove, who profoundly influenced the way I think about psychology and helped set me on the journey of learning and discovery that led to the present project.

Introduction

Psychological assessment involves the evaluation of an individual's standing on one or more psychological constructs of interest. Though psychological assessment data can be gathered from many sources (e.g., interview, psychological test, behavioral observation), the evaluator eventually must combine these data to generate inferences about the individual being assessed. Psychologists involved in assessment tend to base their decision-making (e.g., diagnosis, recommendations) upon informal integration of the assessment data; however, a substantial body of evidence suggests that the decision process is most effective when driven by application of formal statistical prediction models or other mechanical procedures (Ægisdóttir et al., 2006; Grove, Zald, Lebow, Snitz, & Nelson, 2000; Meehl, 1954). On its face, psychological assessment appears a ripe opportunity for applied statistical prediction; many of the data gathered during a psychological assessment (for example, psychological test results) are quantitative in nature or are at least quantifiable. Such data lend themselves particularly well to integration in predictive statistical models, which provide an empirical basis for generating predictions and guiding other decisions and recommendations.

However, the *generalizability* of a statistical prediction model limits its practical utility. That is, such a model is useful only insofar as it allows us to meaningfully predict *new* data, not just the data used to develop the model (Busenmeyer & Wang, 2000). The generalizability

problem is of great practical importance. Despite the apparent superiority of statistical prediction over human judgment, traditional statistical prediction models, such as linear and logistic regression, have met only limited success when validated on external data, particularly when the prediction model includes multiple predictor variables (Browne, 2000; Mosier, 1951). Although human behavior is inherently complex and multifactorial, and is therefore difficult to predict, the problem of limited prediction generalizability also owes substantially to limitations of the statistical methods employed (Babak, 2004; Copas, 1983; Darlington, 1968).

The Case for Statistical Prediction

Meehl (1954) controversially argued that when an appropriate statistical prediction model is available, clinical psychologists should base their decision-making on the output of the model, rather than on their own expert judgment. To support this argument, Meehl (1954) presented a survey of approximately 20 studies in which clinical prediction and statistical prediction were directly compared; in each case, statistical prediction performed as well as, or better than, expert clinical prediction. This finding was counter-intuitive to many clinicians, and a lively debate on the relative merits of clinical and statistical prediction ensued. Nevertheless, subsequent empirical investigations, including two large-scale meta-analyses (Ægisdóttir et al., 2006; Grove, Zald, Lebow, Snitz, & Nelson, 2000) have vindicated Meehl's (1954) position. These studies found that across applied scientific fields (not just within the domain of clinical psychology), experts are rarely able to outperform statistical models when making predictions, even with access to more information. Rather, in most cases, statistical prediction performs as well as, or better than expert judgment. No situations in which a clinician might be expected to systematically outperform a statistical prediction model were identified¹. Human cognitive

¹ However, Meehl (1957) did argue that clinicians have an advantage in identifying "broken leg cases": cases in which special and unusual circumstances (such as breaking one's leg, which could lead to a deviation from normally

limitations appear to be at fault for the underperformance of expert judgment relative to statistical prediction. For example, human beings are prone to cognitive bias, and as such, they have a difficult time objectively weighting various predictor variables when attempting to arrive at an optimal probabilistic judgment.

These findings, though influential, have not led to the proliferation of applied statistical prediction within the field of psychology that Meehl had originally envisioned (Grove & Meehl, 1996; see also Meehl, 1956, 1986). Nevertheless, the demonstrable general superiority of a statistically-oriented approach to prediction and decision-making suggests that development of useful statistical prediction models is a worthwhile pursuit for assessment-oriented psychology researchers. In fact, given that the process of psychological assessment potentially provides a great deal of information that can be used to drive predictions and decision-making with significant real-life consequences, development of decision-guiding statistical models is arguably a central task for the science of psychological assessment (Meehl, 1956). Moreover, because statistical prediction models offer a general advantage over expert human judgment, development and application of such models is particularly consonant with the goals and ethics of clinical psychologists and other applied psychologists who seek to use psychological assessment to improve the well-being of others, as improved prediction will tend to lead to more optimal decision-making and, therefore, to improved outcomes.

Similarly, psychologists are increasingly expected to engage in evidence-based practice (EBP) while operating clinically, using sound empirical research to guide intervention (Anderson, 2006). Accordingly, psychologists conducting assessments in such settings should

predictable behavior) justify ignoring or over-ruling a statistical prediction model. However, he also argued that clear-cut broken leg cases occur very seldomly in applied psychology (perhaps much more seldomly than clinicians believe), and as such, clinicians are better off relying upon statistical prediction the vast majority of the time.

engage in evidence-based assessment (EBA), using empirically-supported methods to optimally meet the needs of evaluatees, service professionals, third-party payers and referral sources, and other affected parties (Anderson, 2006; Hunsley & Mash, 2007). Thus, if formal, applied statistical models offer a potential advantage over the subjective judgment-based prediction methods (e.g., impressionistic combination of assessment data) that currently predominate many areas of applied psychology, then a focus of assessment psychology research should indeed be on the development of statistical procedures that can be applied in practice to aid real-world decision-making. Moreover, in areas in which some limited form of statistical prediction is already in place (for example, in using single-scale cut scores to determine the applicability of empirically-validated interpretive statements), the potential for improvement of prediction through more advanced statistical models merits investigation.

Generalizability of Traditional Statistical Prediction Methods

Single and multiple regression are the most common statistical prediction methods traditionally used in psychological assessment research. Single regression is a bivariate prediction method in which a simple formula is employed to predict the value of one variable (the criterion or outcome variable) based upon the given value of another (the predictor). Multiple regression extends the bivariate model to incorporate data from additional predictor variables. Most commonly, such models are used to predict a continuous (or quasi-continuous) outcome variable using a linear, additive combination of weighted predictors. Logistic regression, a generalization of the linear model designed to predict dichotomous class membership, is also commonly used in psychology research. Although other predictive modeling approaches, such as discriminant function analysis, have also seen a substantial amount of use within assessment psychology, simple regression-based approaches remain among the

most popular and enduring methods for prediction, likely owing at least in part to their ease of use and interpretation. Linear multiple regression models in psychological assessment research (and most other psychology research) are almost always calibrated using an ordinary least squares (OLS) approach, which algorithmically optimizes the regression model by minimizing squared errors of prediction across the range of data used to develop the model.

These predictive modeling techniques are somewhat limited with respect to the generalizability of the models they produce. Although single regression models tend to remain relatively stable across data sets, given appropriate model development data (see, for example, Cohen, Cohen, West, & Aiken, 2013), such a model only accounts for variance shared between two variables. This is unsatisfactory in applied psychology research given that criteria of interest (e.g., psychological constructs, human behavior) are generally complex and may share non-redundant explanatory variance with any number of additional potential predictors. Limiting the information used to generate predictions restricts the amount of predictable information, which in turn limits the validity of predictions generated by bivariate models. Thus, use of multiple prediction appears to offer a significant potential advantage over single regression with regard to creating models of greater general validity because inclusion of multiple predictor variables hypothetically permits the model to account for a greater amount of criterion variance than would inclusion of only a single predictor.

However, this gives rise to a problem often under-appreciated in the psychological sciences. Namely, as the number of predictors in a model increases, conventional OLS techniques for multiple regression increasingly tend to produce unstable predictor weights of limited generalizability (Babiyak, 2004; Browne, 2000; Copas, 1983; Darlington, 1968; Mosier, 1951). As this happens, model fit will generally tend to continue to improve with respect to the

calibration data. However, when such a model is then used to predict new data points for cases not included in the calibration data, model performance tends to be much poorer than one would expect given its good fit to the calibration data. In such a case, the model is said to be *overfit* to the calibration data, meaning that it is unlikely to perform comparably well in any new dataset. The attenuation of goodness-of-fit indices observed when using an overfit model to predict new data has been termed *shrinkage* (Copas, 1983).

Generalizability of statistical predictions from psychological assessment data is further limited by discrepancies between the criterion distributions in the model development sample and external applied samples. For example, when the criterion is dichotomous in nature, and the base rate (prevalence) of the target class in an applied setting is lower than its base rate in the development setting (as is often the case when translating psychology research to real-world application), the predictive power of the statistical model is generally reduced, potentially to the extent that the user is worse off using the model than simply “betting the base rates” (Hunsley & Meyer, 2003; Meehl & Rosen, 1955; Wiggins, 1973). Similarly, if the range of a continuous criterion variable is restricted in the development sample (which typically occurs due to a biased sample selection process), the model’s ability to extrapolate beyond those restricted data will be hindered, and hence its generalizability will be limited (Linn, 1968). The effects of both base rate shifts and range restriction on statistical prediction are well-known to assessment psychologists, though psychologists rarely formally account for these phenomena in their predictions (Linn, Harnisch, & Dunbar, 1981; Meehl & Rosen, 1955).

A Way Forward: Machine Learning

Use of traditional prediction methods in assessment psychology research has in some important ways fallen short of producing generalizable statistical models of the sort needed to

advance applied statistical prediction. This problem is, of course, not limited to the assessment area of psychology. Indeed, some have argued that failure to produce generalizable predictive models is a significant shortcoming for psychology research broadly and that research psychologists in general should be much more focused on producing predictive models that generalize across samples, rather than on evaluating explanatory models based on their fit to a set of calibration data (see Yarkoni & Westfall, 2017). This raises the question of whether traditional statistical methods in the study of psychology can be improved upon or replaced in order to advance generalizable applied prediction. As Yarkoni & Westfall (2017) have suggested, one promising avenue for such improvement is through adopting the methods of a successful field in which generalizable statistical prediction is a central focus: machine learning.

Machine learning, also sometimes called statistical learning, is a branch of applied statistics falling under the broader umbrella of data science. Modern machine learning is often associated with computer science and complex computer learning problems such as image recognition and artificial intelligence. However, virtually any statistical prediction method in modern use potentially falls within the domain of machine learning. Indeed, linear and logistic regression are some of the most commonly used machine learning techniques (James, Witten, Hastie, & Tibshirani, 2014). However, machine learning also includes many statistical methods not commonly employed by psychologists but potentially applicable to prediction problems in psychology. Furthermore, machine learning's approach to statistics notably differs from the approaches traditionally used by psychology researchers in that machine learning places a premium on the generalizability of findings. That is, machine learning differs from conventional research methods in psychology with regard to model evaluation. In psychology research, models are often positively regarded if they fit well to the data used to calibrate the model, but in

machine learning, the gold standard for model performance is the model's goodness of fit to validation data (i.e., data not used to develop the model; Efron & Hastie, 2016; Hastie, Tibshirani, & Friedman, 2009; James et al., 2014; Yarkoni & Westfall, 2017). To this end, machine learning's toolbox includes not only a variety of methods to model data, but also techniques to fairly estimate and evaluate generalizable model performance.

Machine Learning: A Brief Introduction

As just mentioned, machine learning is a branch of applied statistics and data science. The boundaries of what is considered machine learning are not clearly defined, as the term "machine learning" represents as much an approach to statistics as it does a distinct field. As such, it both incorporates methods from, and disseminates methods to many areas of applied statistics. Machine learning has become increasingly popular in the age of modern computation and "big data" (Efron & Tibshirani, 2016; Hastie et al., 2009; James et al, 2014). Many of its techniques were first developed decades ago; however, recent advances in computing hardware and software have made their widespread applications feasible. Furthermore, data are now available more readily and in much greater quantities than ever before, providing many new opportunities to develop, apply, and test predictive models, though some sophistication is often required to make sense of these large and often complex datasets. Indeed, many machine learning procedures are specifically designed to deal with modeling problems inherent in large sets of complex data (for example, discriminating "true" predictors from spurious ones, fitting a predictive model when the number of predictors is greater than the number of observations, or selecting a final predictive model from among a very large number of competing alternative

models; James et al., 2014; Kuhn & Johnson, 2013). The purpose of this section is to provide a conceptual overview of key machine learning concepts².

Machine learning terminology.

This section introduces a number of terms commonly used in machine learning. Many of these terms are also used in assessment psychology; however, common terms may differ in definition and usage between the fields of machine learning and psychological assessment. Additionally, these two areas share a number of concepts that are given different labels depending on the field. After these terms are introduced, the (generally more precise) machine learning language conventions will be adhered to whenever possible for the remainder of the paper.

Training and test data. Simply put, *training data* are any data used to calibrate a statistical model, whereas *test data* are any data used to evaluate a statistical model. In the machine learning approach, no overlap exists, under most circumstances, between training and test data, though both training and test data may represent subsets of a larger sample of data. In psychology research, training data are often called “development” or “calibration” data, and test data are often referred to as “validation” data. However, both of these psychology terms are ambiguous. For example, “development” can refer to development of a test, a prediction model, or a theoretical model, and there are many kinds of validation, as discussed below. Describing a particular set of data as “training” or “test” identifies its use with statistical modeling while also describing its role within the model calibration and evaluation processes.

Validation and cross-validation. In assessment psychology, the term “validation” has many meanings. For example, although validation can refer to evaluation of a statistical model,

² See “Method,” below, for a technical discussion of specific techniques used in the present study.

as is relevant in the present discussion, it can also refer to establishment of various forms of theoretical or psychometric validity (e.g., construct, content, criterion). In machine learning, *validation* refers specifically to evaluation of a statistical model using test data. Similarly, “cross-validation” has a more precise meaning in machine learning than in psychology research. In psychology research, cross-validation refers to any evaluation of a model or measure outside the training data (often using an entirely different sample). However, in machine learning, *cross-validation* usually refers specifically to a family of resampling procedures used in model selection and evaluation (Kohavi, 1995), described below (see “Method”). To minimize ambiguity across the remainder of the paper, in addition to using the machine learning definitions for validation and cross-validation, the author will use the term *external validation* to describe the specific case in which a model is validated on a test dataset entirely distinct from the original data (i.e., not merely a holdout set of the original sample).

Tuning variables. A *tuning variable* or *tuning parameter* is a variable used in the model training process to calibrate the model; however, tuning variables are not generally used to generate predictions directly. Tuning variables are often pre-specified by the researcher or are selected by comparing cross-validated performances of competing models along various values of the tuning variable. For example, regularized regression (described later) uses a shrinkage penalty, a tuning variable that controls the extent to which the regression beta weights are shrunk toward zero when those weights are calibrated. A researcher developing a regularized model could simply specify the value of the shrinkage penalty. More commonly, however, the researcher will allow the statistical software package to determine the empirically optimal value of the shrinkage penalty by developing multiple “versions” of the model, each with different values for the shrinkage penalty (and therefore with different parameter estimates for the beta

weights), then selecting the version of the model with the highest cross-validated performance as the “final” version of the model (i.e., the one that whose parameters will be used).

Considerations in prediction method selection and model specification. In selecting a predictive modeling approach, researchers are faced with a multitude of potentially viable alternative methods. However, an informed researcher must also consider many factors in order to select a method that is not only appropriate to the data and to the function of the model but also likely to produce sufficiently generalizable results. In many cases, the approach the researcher selects represents a compromise between competing goals of research and application (for example, model interpretability vs. predictive accuracy).

Regression vs. classification. In machine learning, the regression-classification distinction refers to the nature of the criterion variable involved in the prediction task. *Regression* in this sense refers to prediction of a continuous or quasi-continuous criterion, whereas *classification* refers to prediction of discrete group membership (Hastie et al., 2009; James et al., 2014; Kuhn & Johnson, 2013). To a certain extent, this distinction is artificial, as mechanically similar methods are often applied to both regression and classification problems. Nevertheless, an informed researcher must be aware of the distinction in order to select a modeling approach that is validly applicable to the prediction problem and produces sensible output of an appropriate form.

Flexibility, interpretability, and the bias-variance trade-off. *Flexibility* is the extent to which a predictive model is able to account for variations in distributions of data. *Interpretability* is the extent to which an individual examining the trained model is able to understand how the model produces predictions as a function of its predictors (see Hastie et al., 2009). Flexibility and interpretability are not inherently incompatible; however, highly flexible

modeling approaches tend to be more difficult to interpret than more rigid approaches (James et al., 2014). When accuracy of prediction of external data is the primary research goal, interpretability may not be a major consideration in selecting a modeling approach; however, researchers interested in forming explanatory theoretical inferences about the predictors may wish to select a more interpretable model over one that has good predictive power but limited interpretability. In addition, the standards of a given domain of applied prediction may weigh against use of an uninterpretable model. A naïve researcher focused solely on prediction might be tempted to consistently prefer a flexible modeling approach to a more rigid one, as a flexible approach might appear on its surface better suited to account for the nuanced associations often observed in real-world data. However, to maximize generalizable predictive power, a sophisticated researcher must also consider the relationship between model flexibility and the bias-variance trade-off, which is discussed next.

Understanding model bias, model variance, and the trade-off between them is a fundamental requirement for the knowledgeable development and evaluation of generalizable predictive models. Virtually all prediction models applied to test data produce some errors of prediction. Error due to *model bias* is the amount that the model errs, on average, relative to its expected (i.e., predicted) values. Model bias error tends to result from reducing a complex process to a simplified statistical approximation. Error due to *model variance*, on the other hand, is the error in prediction that occurs due to variability in the training data (i.e., sampling error). The parameter values of a model with high error due to variance will tend to change substantially if the model is retrained on different data.

Models with high bias are often said to be *underfit*. An underfit model does not capture important nuances of the “true” underlying function giving rise to the observed data. For

example, if a linear model is used to predict data produced by an underlying quadratic function, the linear model will underfit the data and will be systematically biased. On the other hand, a model with high variance is often *overfit*, meaning that its form reflects variability in the training data that is not actually representative of the underlying function that gave rise to those data. For example, if a regression model with many higher-order polynomial terms is trained on a set of data derived from an underlying linear function, that model will likely fit the training data very well. However, when used to predict external data, it will perform very poorly because the complex polynomial form of the model has been strongly influenced by error variance in the training data and is a poor representation of the true linear function. In the interest of model generalizability, we would ideally apply a model that is neither underfit nor overfit, one that does a good job of describing the “true” probabilistic process that gives rise to new observations, rather than one that simply reflects an arbitrarily simplified model form or noise in the training data.

The estimated error rate of a prediction model can be operationalized as the estimated mean square error, or $E[MSE]$, which can be defined as

$$E[MSE] = Model\ Bias\ Error^2 + Model\ Variance\ Error + Irreducible\ Error , \quad (1)$$

where irreducible error is inherent variability in the criterion variable that cannot be eliminated or reduced, even if the form of the true function is perfectly described, because it is introduced by variables that are not directly represented in the data, such as measurement error (see James et al., 2014). Thus, researchers wish to minimize both model bias and model variance in order to maximize prediction accuracy with respect to new data. Unfortunately, bias and variance are often at odds with one another. A model with low bias must be flexible enough to meaningfully approximate the form of the “true” underlying function; however, as model flexibility increases,

so does the risk of overfitting the data. The key to producing a generalizable model is to strike an optimal balance in the *bias-variance trade-off*, which requires an informed approach to model evaluation and selection (Efron & Tibshirani, 2016; Friedman, 1997; Hastie et al., 2009; James et al., 2014).

Complexity vs. parsimony. A complex model is one that includes many predictors, or *features*; in contrast, a parsimonious model includes relatively few. The complexity of a model relates to the bias-variance trade-off. A parsimonious model with few predictors tends toward high model bias but low model variance. As complexity increases (i.e., as more features are added to the model), variance increases while bias tends to decrease. However, addition of a feature only decreases prediction error if the additional feature accounts for a sufficient amount of additional variance in the criterion. Adding features that have no predictive power serves only to increase model variance, leading to overfitting and reducing the generalizability of the model (Miller, 2002). In addition to limiting model variance, parsimonious models tend to be more interpretable than more complex ones. For these reasons, statisticians tend to prefer parsimonious models over more complex alternatives, even when model performance criteria suggest they have equivalent predictive power (see, e.g., Hastie et al., 2009; James et al., 2014; Miller, 2002). However, the most appropriate level of complexity for a model is not always clear; in fact, identifying the optimal number of features to include in a model is a common task in statistical learning (see “Feature Selection,” below). Prediction methods that tend to produce effectively uninterpretable models are often referred to as “black box” methods.

Psychology researchers tend to place a premium on model parsimony in predictive modeling, strongly preferring simple to more complex models. This is especially the case when the more complex models under consideration include predictors whose inclusion cannot be

justified *a priori* by a strong theoretical association between the predictor and criterion (see “Feature Selection,” below). However, the practical advantages and disadvantages of parsimony in applied prediction in psychology is not well-understood, as models of varying complexity are rarely empirically compared to one another, and even when they are compared empirically, such models are rarely evaluated fairly with regard to their generalizable predictive power.

Model evaluation and selection. The distinction between training and test data is key to understanding the evaluation of a predictive model’s generalizability. As discussed above, in psychological research, training data performance is often misleadingly put forth as the expected future performance of a predictive model (Babyak, 2004; Yarkoni & Westfall, 2017). However, because a predictive model’s parameters are optimized to maximize the model’s fit to the training data, such a model will virtually always fit its training data better than it will any test data, including the data produced in the real-world settings in which professionals may wish to apply the model. Furthermore, evaluating the potential generalizability of a candidate predictive model requires some criterion of model performance. Many such criteria have been proposed and put into practice over the years, each with advantages and limitations. Thus, evaluation of the predictive performance of a model involves consideration of the data on which the model is tested as well as evaluation of the model performance criterion. The gold standard of model performance is external validation, ideally using multiple external samples, though resampling techniques (e.g., *k*-fold cross-validation) and penalized training data performance criteria (e.g., adjusted R^2) are sometimes used when external data are not available (Hastie et al., 2009, James et al., 2014; Kuhn & Johnson, 2013).

Selection of an appropriate model performance criterion requires consideration of the nature of the model and the outcome variable. For example, regression and classification tasks

generally require different evaluation criteria from one another; in the instances in which the same performance criterion can be used for both types of prediction task, the criterion is calculated differently for each. Some model performance criteria are especially useful for comparing models of different types within the regression and classification frameworks. For example, test set mean square error (MSE), calculated simply by taking the squared residuals of the outcome variable in a set of test data, allows fair overall comparisons of the performances of virtually any regression models evaluated against identical test data. Similarly, overall classification accuracy provides a basis for comparing nearly any classification models of different types to one another. Specific model evaluation criteria relevant to the present study are discussed in greater detail below (see “Method”).

Model evaluation is also an important part of model selection. It is used when the researcher must select a final model from a number of competing alternatives. Model selection is often performed algorithmically. Given a loss function (a function that the optimizing algorithm seeks to minimize) representing some form of model performance (for example, minimizing MSE on cross-validated data), a statistical software package can generate and compare many different models. Common uses of algorithmic model evaluation and selection in machine learning include feature selection (selecting a subset of possible predictors to serve in a final model) and selection of values for tuning parameters.

Feature selection. As noted above, feature selection is the process of selecting, from the pool of all possible candidate predictors, those predictors that will be included in the final predictive model. This becomes particularly important in research situations in which the number of predictors is very large; in fact, many conventional modeling techniques (e.g., OLS regression) cannot estimate model parameters if the number of predictors exceeds the number of

observations. Thus, in such situations, feature selection (or regularization, discussed below) is not only advisable, it is necessary. Although psychology researchers generally advocate for theory-driven feature selection, many mechanical alternatives are available and are commonly used in machine learning.

Appropriate use of mechanical feature selection requires an understanding of both the bias-variance trade-off and model evaluation, as feature selection often involves comparison of many competing models of varying complexities. The most comprehensive form of feature selection is all-subsets selection, in which literally all possible combinations of predictors are considered before selecting the highest-performing model (according to some model performance criterion). However, the computational demand of the all-subsets algorithm increases exponentially as the number of candidate predictors increases, and beyond a modest number of potential predictors (typically around 40), execution of the all-subsets routine is not computationally feasible for most researchers. For this reason, other mechanical feature selection processes perform a more guided search of feature space without requiring calculation of performance for every possible model. These methods range from simple stepwise procedures that systematically add and/or subtract variables to more complex search heuristics such as genetic algorithms (Michalewicz, 1994). In general, adding features to a model tends to increase model variance; thus, as model complexity increases, so too does the risk of overfitting. The primary challenge of mechanical feature selection, then, is to ensure that spurious features are not selected. In other words, to the greatest extent possible, features should not reflect noise in the training data, but rather should contribute meaningfully to the model's generalizable predictive power. A secondary challenge is proper evaluation of the generalizable predictive performance of candidate models. For this reason, appropriate evaluation of candidate

predictive models developed using mechanical feature selection requires careful use of validation data or penalized model evaluation criteria (Ambler, Brady, & Royston, 2002; Babyak, 2004; Berk, 2008; Hastie et al., 2009; Kuhn & Johnson, 2013; Miller, 2002).

Special mention must be made of the status of automated feature selection methods in the study of psychology. Although psychology researchers are generally familiar with basic stepwise feature selection methods, these techniques have fallen out of favor due to concerns regarding their limited generalizability. In fact, at least one psychological assessment journal completely banned studies employing stepwise selection (Thompson, 1995). Fears regarding the generalizability of stepwise methods are not entirely unwarranted; indeed, empirical research has shown that such methods tend to overfit their training data and to include spurious features in their final model (Derksen & Keselman, 1992; Steyerberg, 2009). Psychology researchers' concerns regarding the validity of stepwise selection appears to have expanded to suspicion of all automated feature selection techniques. Nevertheless, machine learning's toolbox includes a robust suite of techniques potentially useful for both automated feature selection and evaluation of feature-selected models, and many of those techniques include elements intended to mitigate concerns such as overfitting.

The Present Study

Thus far, the author has endeavored to demonstrate three primary points. First, assessment psychologists have compelling reasons to develop and apply statistical prediction models to aid decision-making in real-world contexts. Second, for a variety of methodological reasons, traditional approaches to statistical prediction in traditional psychology research tend to produce models of limited or questionable generalizability; moreover, the generalizable predictive power of these models is rarely evaluated fairly. Third and finally, the field of

machine learning, a branch of applied statistics, employs an array of principles and methods that could potentially be incorporated into psychological assessment research in order to improve both statistical prediction and model evaluation with respect to generalizability.

In the present study, the author aims to evaluate the extent to which statistical prediction and model evaluation in psychological assessment may be improved by application of machine learning concepts and techniques. The author will compare traditional prediction methods currently in wide use in psychology research, such as ordinary least squares (OLS) linear and logistic regression, to alternative modern prediction techniques commonly used in the field of machine learning. In accordance with best-practice machine learning principles, models will be evaluated with an emphasis on test data performance, rather than on training data performance. Three broad classes of techniques will be considered: regression, classification, and feature selection. In all analyses, test score data obtained from a broadband measure of personality and psychopathology, the Minnesota Multiphasic Personality Inventory-2 Restructured Form (MMPI-2-RF; Ben-Porath & Tellegen, 2008), will serve as predictors, whereas various clinician-rated clinical constructs of interest (mental status exam variables, psychiatric diagnoses) will serve as prediction criteria. Results of the present study will inform responses to four research questions related to the central issue of the generalizability of statistical predictions in psychological assessment. First, to what extent do the methods under consideration tend to produce generalizable predictions? Second, how well do the methods under consideration compare with one another with respect to prediction generalizability? Third, to what extent does a model benefit from having more or less predictive information (i.e., how should we value parsimony in applied prediction)? Fourth and finally, what is the most effective way to approach feature selection when attempting to maximize generalizable predictive power in psychological

assessment? Responses to these questions, informed by the results of this empirical investigation, should provide important guidance in the selection of modeling approaches for assessment psychologists who wish to develop useful, generalizable predictive models.

Psychology and machine learning are vast fields, and testing more than a small subset of their respective techniques would be unfeasible. For the purposes of the present study, “traditional” techniques are drawn from the pool of statistical prediction methods judged by the author to be most commonly used in psychological assessment research (for example, conventional OLS linear regression). Machine learning techniques in the present study are selected based not only on their apparent rates of usage and acceptance in machine learning and other areas of applied statistics, but also on their demonstrable “off-the-shelf” success. That is, the machine learning techniques used in the present study are selected based on their ability to perform competitively in terms of predictive performance (relative to other candidate techniques) without placing a great burden of technical knowledge or excessive preparatory work upon the researcher developing the model. The requirement that these modeling techniques have a history of performing well “off-the-shelf” is stipulated for two reasons. First, this property mitigates the probability that apparent model performance in the present study reflects a misuse of the modeling technique, to the extent that the simplicity of these methods’ applications precludes user error. Second, demonstration of generalizable prediction using a new technique that is easy to apply increases the likelihood that the technique will be adopted by others (relative, say, to a demonstration using a model producing similar performance but requiring greater technical knowledge on the part of the researcher). Selection of machine learning techniques is guided both by leading texts in the area (Hastie et al., 2009; James et al., 2014; Kuhn & Johnson, 2013) and by past empirical comparative studies outside the field of psychology (particularly Caruana

& Niculescu-Mizil, 2006; and Fernández-Delgado, Cernadas, Barro, & Amorim, 2014). Specific selected techniques are described in detail under “Data Analysis Strategy,” below.

Method

Participants

The present study employs archival data originally collected at three separate clinical sites. Two of these samples represent psychiatric inpatient populations: one from a large metropolitan county hospital in the American Midwest, the other from a VA hospital in the same geographic region. The third sample was drawn from a Midwestern outpatient community mental health center. Participants producing invalid MMPI-2-RF protocols according to standard MMPI-2-RF protocol validity criteria ($CNS \geq 18$, $VRIN-r \geq 80$, $TRIN-r \geq 80$, $F-r \geq 120$, $Fp-r \geq 100$; Ben-Porath & Tellegen, 2008) were excluded from analysis. Table 1 displays exclusion rates and descriptive demographic data for the participants retained from these three samples.

Table 1

Sample Exclusion Rates and Demographic Characteristics

Characteristic	Psychiatric Inpatients, County	Psychiatric Inpatients, VA	Community Outpatients
Percent excluded ^a	31%	31%	12%
Final <i>N</i>	1,050	972	895
Gender	57.0% male	93.3% male	61.2% female
Mean Age	34.0	48.1	33.0
Mean Years of Education	9.3	9.7	12.5

Note. ^aPercent excluded due to MMPI-2-RF protocol invalidity.

Measures

Minnesota Multiphasic Personality Inventory-2 Restructured Form (MMPI-2-RF).

All retained participants produced valid protocols for the MMPI-2-RF, a 338-item, 51-scale broadband measure of personality and psychopathology³. In addition to appraising threats to protocol validity (e.g., inconsistent responding, overreporting, underreporting), the MMPI-2-RF's scales measure substantive content spanning multiple domains of psychopathology, including internalizing dysfunction, externalizing dysfunction, thought dysfunction, interpersonal dysfunction, and somatization. The MMPI-2-RF has been the subject of numerous studies evaluating its psychometric properties (see Ben-Porath, 2012a, 2012b; Tellegen & Ben-Porath, 2008). Its scales have been found to be sufficiently reliable and to have good convergent and discriminant validity with respect to empirical associations with extra-test criteria (Tellegen & Ben-Porath, 2008). Extensive MMPI-2-RF reliability and validity coefficients for the samples used in the present study are reported in Tellegen & Ben-Porath (2008). Internal consistency (Cronbach's α) values for the MMPI-2-RF substantive scales range from .60-.95 ($M = .77$) for the county hospital sample, .50-.92 ($M = .75$) for the VA hospital sample, and .57-.93 ($M = .75$) for the community outpatient sample.

Record review form scales. Research assistants completed identical record review forms (RRFs) for participants in the county hospital inpatient and VA inpatient samples. These RRFs included dichotomous mental status exam items indicating the presence or absence of specific patient symptoms or characteristics at the time of intake. In previous studies (see, for example, the MMPI-2-RF Technical Manual; Tellegen & Ben-Porath, 2008), these mental status exam items were aggregated into scales measuring clinical constructs of interest. Four such

³ Participants were originally administered the Minnesota Multiphasic Personality Inventory-2 (MMPI-2; Butcher, Dahlstrom, Graham, Tellegen, & Kaemmer, 1989), a 567-item measure, rather than the MMPI-2-RF. All items from the MMPI-2-RF are found in the MMPI-2 item pool; thus, the MMPI-2-RF can be scored from any MMPI-2 protocol. As such, all MMPI-2-RF protocols used in the present study were rescored from MMPI-2 protocols.

specific RRF scales (Depression, Substance Abuse, Delusions, and Pain) are used in the present study as prediction criteria in regression analyses; any other previously created RRF scales are not used. Intra-class correlation (ICC) estimates of reliability for these scales range from .69 (Pain) to .91 (Delusions), with a mean ICC of .85 (Tellegen et al., 2003). Table 2 lists the dichotomous items comprising each of the RRF scales used in the present study. Items on these scales are scored 1 if the clinical feature is present or 0 if the feature is not present. Means and standard deviations for the RRF scales, separated by sample, are presented in Table 3. Visualizations of the distributions of the RRF scales are presented in Figure 1 for the county hospital sample and in Figure 2 for the VA medical center sample.

Table 2

Prediction Criteria and Their Compositions

Regression Criteria		Classification Criteria	
RRF Scale	Scale Items	Diagnostic Group	Included Diagnoses
Depression	Worthlessness Guilt Helpless or Hopeless Tearfulness Anhedonia	Depressive Disorder	Major Depressive Disorder Dysthymia Depressive Disorder NOS
Substance Abuse	History of Substance Abuse Alcohol Polysubstance Hallucinogens Opioids Benzodiazepines Cocaine Marijuana	Substance Use Disorder	Substance Abuse Substance Dependence
Pain	Chronic Pain Backache Headache Muscle Ache	Somatic Disorder	Hypochondriasis Somatization Disorder Somatoform Disorder Conversion Disorder
Delusions	Delusions Ideas of Reference Delusions of Reference Persecutory Delusions Paranoid/Suspicious	Thought Disorder	Schizophrenia Schizoaffective Disorder Schizophreniform Disorder Psychotic Disorder NOS

Note. RRF = Record Review Form.

Table 3

Distributional Characteristics of Prediction Criteria

Criterion	County Hospital	VA Medical Center	Community Outpatient
<i>Record Review Form Scales: Mean (SD)</i>			
Depressed	1.90 (1.67)	.93 (1.09)	
Substance Abuse	2.15 (2.03)	1.85 (1.82)	
Delusions	.71 (1.14)	.43 (.89)	
Pain	.25 (.70)	.34 (.77)	
<i>Diagnostic Groups: Base Rates</i>			
Depressive Disorder	.46	.39	.24
Substance Use Disorder	.44	.45	.14
Thought Disorder	.17	.14	.02
Somatic Disorder	.01	.02	<.01

Figure 1

Record Review Form Scale Score Distributions: County Hospital

Figure 2

Record Review Form Scale Score Distributions: VA Medical Center

Clinical diagnoses. Clinical psychiatric diagnoses were coded for all participants in the present study. Diagnoses were assigned by the participants' treating clinicians according to the diagnostic criteria contained in the Diagnostic and Statistical Manual III-Revised (DSM-III-R; American Psychiatric Association [APA], 1987) or Diagnostic and Statistical Manual-IV (DSM-IV; APA, 1994). Although operationalized in the DSM as discrete categorical diagnoses, a large body of research indicates that many psychological disorders are associated with one another through higher-order dimensional factors under which similar disorders are grouped (see Kotov et al., 2017). Thus, in the present study, four diagnostic groups are created to represent various major areas of psychopathology: Depressive Disorder, Substance Use Disorder, Thought Disorder, and Somatic Disorder. A participant is considered a member of one of these groups if the participant's chart notes a formal diagnosis of any of the disorders comprising the group.

Membership statuses in these diagnostic groups are used as prediction criteria in the present study's classification method analyses. The specific DSM diagnoses defining each group are listed in Table 2. The base rates for each diagnostic group, separated by sample, are displayed in Table 3.

Data Analysis Strategy

General approach. The availability of predictive information in these models is systematically varied in order to explore the impact of different approaches to model parsimony and feature selection on model generalizability. Thus, in addition to being separated into regression and classification analysis branches, analyses are also organized into three groups representing different approaches to predictor selection. In the first group of analyses, only scales from the set of Restructured Clinical (RC) Scales are used as predictors (representing nine of the 42 MMPI-2-RF substantive scales). This scale set, arguably the most essential to interpretation of the MMPI-2-RF, covers a broad range of clinically important constructs. The RC Scales include RCd (Demoralization), RC1 (Somatic Complaints), RC2 (Low Positive Emotions), RC3 (Cynicism), RC4 (Antisocial Behavior), RC6 (Ideas of Persecution), RC7 (Dysfunctional Negative Emotions), RC8 (Aberrant Experiences), and RC9 (Hypomanic Activation). In the second set of analyses, the full pool of all 42 substantive scales are used as predictors for each model. In the third and final analysis group, all 42 substantive scales are used to form a pool of *potential* predictors; however, the predictors for the final models in this group are selected using various feature selection approaches. The nine MMPI-2-RF validity scales are not used as predictors in any analyses. The specific modeling approaches used in each group of analyses are listed in Table 4.

Table 4

Modeling Techniques Used in Present Study

Predictors	Regression Methods	Classification Methods
RC Scales	OLS linear regression	MLE logistic regression
	Negative binomial regression	Ridge logistic regression
	Ridge linear regression	Random forests
	Random forests	Support vector machines
	Support vector regression	
All Substantive Scales	OLS linear regression	MLE logistic regression
	Ridge linear regression	Ridge logistic regression
	Random forests	Random forests
	Support vector regression	Support vector machines
Feature Selection	Single-predictor selection	Single-predictor selection
	Manual selection	Manual selection
	Forward selection	Forward selection
	Backward deletion	Backward deletion
	Forward-and-backward selection	Forward-and-backward selection
	Lasso	Lasso

Note. OLS = Ordinary Least Squares; MLE = Maximum Likelihood Estimation.

Within each set of analyses, multiple modeling strategies are employed using the same starting set of predictors and the same prediction criteria. Use of multiple modeling strategies with the same training data and external validation data provides a basis for fair comparison of model performance. Multiple prediction criteria are employed for each modeling strategy in order to provide additional points of comparison. For all analyses, the county hospital inpatient sample is used for model training. For regression analyses, the VA hospital inpatient sample is used for external validation (with the RRF scales serving as prediction criteria), and for classification analyses, both the VA hospital sample and the community outpatient sample are used for external validation (with diagnostic groups as prediction criteria). Feature selection

analyses employ both continuous and categorical prediction criteria (i.e., both the RRF scales and diagnostic groups), with external validation conducted on any applicable data. The specific modeling techniques employed in the present study are described in the following sections.

Training and test data performance metrics are provided for all predictive models. In all cases, this includes model performance on both the full training set and performance on external validation data. Additionally, whenever possible, k -fold cross-validation (Kohavi, 1995) performance is provided as well. k -fold cross-validation, a resampling procedure, is commonly used in machine learning as part of model development and evaluation. In k -fold cross-validation, the training data are randomly divided into k partitions (or *folds*) of approximately equal size. One data fold is designated the test set, and the remaining folds are combined to form the training set. The model is then trained using the training fold set, and the newly trained model's performance on the holdout test fold is recorded. Then, another fold is designated the test set, and the previous test fold is added back to the training data for the next round of model training. This process repeats until the model has been evaluated on each test fold, then predictive performance across all test folds is averaged. In this way, k -fold cross-validation provides a relatively unbiased estimate of model performance, as test data are always excluded from model training. Many machine learning modeling strategies use k -fold cross-validation not only to estimate generalizable performance of the final model, but also to calibrate tuning variables (i.e., variables used not to directly generate predictions but to control aspects of the model optimization process, as described previously). To accomplish this, the value of the tuning variable is systematically adjusted, while the cross-validated performance of the corresponding model is evaluated at each value of the tuning variable. Then, the tuning variable for the training of the final model is set to the value associated with the best cross-validated

performance. In k -fold cross-validation, the variable k can be set to any positive integer, up to the number of observations in the training set (a special case of k -fold cross validation known as leave-one-out [LOO] cross-validation), though typically k is set to a smaller number, usually 10. The present study employs 10-fold cross-validation for cross-validated model evaluation and automated tuning variable calibration, unless otherwise specified.

One challenge in comparing different types of predictive models is selection of an appropriate model evaluation criterion, as some evaluation criteria can only be used for certain types of models. For example, R^2 , the so-called coefficient of determination, is commonly used in psychological research to describe the performance of linear regression models. However, the legitimacy of applying this metric, a measure of linear model fit, to nonlinear models is controversial and is contraindicated by some empirical research (see Spiess & Neumeyer, 2010). Machine learning routinely employs a number of model performance criteria, some of which can be used to evaluate models of many different forms. In the present study, the root mean squared error (RMSE), a measure of absolute model fit, is used to evaluate all regression models⁴, and R^2 values are provided where appropriate. Classification models are evaluated using both overall classification accuracy, evaluating performance of the model as a dichotomous classifier, and area under the curve (AUC) evaluating discriminative performance of the model on the basis of its outputted probabilities⁵.

⁴ RMSE is calculated by taking the root of the mean square error (MSE; the mean of all residuals for the model outcome variable). As a measure of error, RMSE is a “badness-of-fit” metric, with higher values indicating poorer model performance. RMSE can be interpreted as the average error of prediction for the model, in units of the outcome variable.

⁵ Classification accuracy and AUC do not always lead to identical conclusions about comparative model performance. Generally speaking, classification accuracy is preferable when class balance is approximately equal (i.e., the base rate is around .5), and AUC is preferable when classes are highly imbalanced (i.e., the base rate is extremely high or extremely low). In addition, some modeling approaches (most relevantly including random forests) are known to perform well as classifiers and yet to produce poorly calibrated probabilities.

All analyses are carried out in the R software environment (R Core Team, 2015). Data organization and visualization is augmented through use of the ‘tidyverse’ package (Wickham, 2017). Regularized regression is performed using the ‘glmnet’ package (Friedman, Hastie, & Tibshirani, 2009). The ‘MASS’ package (Ripley, Venables, Bates, Hornik, Gebhardt, & Firth, 2013) is used to create negative binomial regression models and to construct stepwise-type models (through the stepAIC function). Random forests are generated using the ‘randomForest’ package (Liaw & Wiener, 2002). Support vector regression and support vector machine models are courtesy of the ‘kernlab’ package (Karatzoglou, Smola, Hornik, & Zeileis, 2004). Wherever possible, predictive models are passed through the ‘caret’ package (Kuhn, 2008), which provides a generalized framework for training and testing predictive models. The ‘pROC’ package (Robin et al., 2011) is used for some ROC analyses. R code for these analyses is presented as an appendix at the end of this document.

Regression methods.

Ordinary least squares linear regression. Ordinary least squares (OLS) linear regression is well-known to researchers in both psychological assessment and machine learning. In OLS linear regression, a linear combination of predictor weights is calibrated to minimize the residual sum of squares (RSS) in the training data. RSS is defined as

$$RSS = \sum_{i=1}^n (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - \dots - \hat{\beta}_p x_{ip})^2, \quad (2)$$

where i is a particular observation, $\hat{\beta}_0$ is the intercept, and $\hat{\beta}_p$ is the regression weight for the p th predictor (James et al., 2014). Linear regression models produce a prediction formula in the following form:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \dots + \hat{\beta}_p x_{ip}, \quad (3)$$

where $\hat{y}$ is the estimated value of the outcome variable.

Negative binomial regression. Though not necessarily a “traditional” prediction strategy in psychological assessment, count regression methods such as negative binomial regression have gained traction in the field in recent years. Negative binomial regression and other count regression techniques are typically used to model *count data*, which take the form of nonnegative integers and typically follow non-normal distributions (e.g., the negative binomial distribution or Poisson distribution; see Hilbe, 2011, 2014). Researchers in psychological assessment have recently suggested that count regression methods are more appropriate for modeling clinical personality data, which often follow count distributions, than the more common OLS linear regression approach (see Wright, Pincus, & Lenzenweger, 2012). Negative binomial regression was selected for inclusion in the present study’s analyses so that this assertion can be fairly tested. Negative binomial regression produces a predictive model of the form

$$\log(\hat{y}) = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}, \quad (4)$$

where

$$\hat{y} = e^{\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \cdots + \hat{\beta}_p x_{ip}}. \quad (5)$$

Ridge linear regression. Ridge regression (Hoerl & Kennard, 1970a, 1970b, 1980) is a *regularized* form of regression. Regularization, a penalizing procedure for constraining the size of prediction model parameter estimates, was designed to improve model generalizability. To understand how the ridge penalty works in linear regression, consider the manner in which OLS (unregularized) linear regression models are optimized: the optimal model is the one that minimizes the RSS (Equation 2) in the training data. When such a model has many predictors, when multicollinearity is high, when the training data are “noisy” (i.e., when much of the variance in the criterion variable is attributable to error), and/or when the number of observations

is low, the model will tend to have high error due to variance and will tend to overfit the training data, reflected in large, unstable predictor weight estimates (Babak, 2004; Browne, 2000; Copas, 1983; Mosier, 1951; Yarkoni & Westfall, 2017).

Ridge regression aims to reduce error due to model variance by introducing a penalty into the model optimization process. Thus, the ridge algorithm seeks to optimize not the RSS, but rather the RSS plus the ridge penalty:

$$RSS + \lambda \sum \beta_p^2, \quad (6)$$

where $\lambda \geq 0$ is a tuning parameter and $\sum \beta_p^2$ is the sum of squared beta weights for all predictors. When λ is set to zero, ridge regression is equivalent to ordinary least squares. However, as λ increases, the optimization function is increasingly penalized for large beta weight estimates. This has the effect of “shrinking” beta weights toward zero, producing conservative but relatively stable weight estimates that tend to slightly increase model bias but significantly decrease model variance⁶. Relative to OLS regression, beta weight estimates produced by ridge regression are fairly robust against perturbations of, and omissions in, the training data (Vinod, 1978). When λ has been well-selected, ridge regression is capable of outperforming OLS regression in predicting test data, sometimes substantially so (Harrell, 2015; Hoerl & Kennard, 1980; James et al., 2014). Ridge regression is particularly useful when the sample size is small or when the signal-to-noise ratio in the data is large (Harrell, 2015). A potential limitation of ridge regression is that its conservative parameter estimates may reduce prediction accuracy for extreme cases, even if they increase the model’s overall accuracy (Efron & Hastie, 2016). Use of the ridge penalty has been extended to problems beyond linear regression; for example, it can

⁶ For this reason, regularization methods are also often called “shrinkage” methods.

be used to potentially improve classification via logistic regression (Le Cessie & van Houwelingen, 1992; see also discussion of ridge logistic regression, below).

Random forests regression. Random forests (Breiman, 2001) are a particularly successful and popular innovation in the field of machine learning. The random forests algorithm is a form of *bootstrap aggregation* (also known as *bagging*). In bagging, the training data are resampled many times via bootstrapping, and a predictive model is fit to each bootstrapped replication of the training data; these models are then combined into an *ensemble*⁷. When a prediction for a new observation is needed, each model in the ensemble offers a prediction, then the predictions of the constituent models are aggregated into a single prediction for the entire ensemble. This aggregation typically occurs via averaging for regression problems or consensus voting for classification problems. Although bagged ensembles are “black box” methods (because they are difficult to interpret due to the large number of models comprising each ensemble), they tend to offer an advantage in prediction because the aggregation process reduces prediction error due to model variance.

The constituent model of the random forests ensemble is a form of classification and regression tree (CART; Breiman, Friedman, Olshen, & Stone, 1984), a flexible and nonparametric decision tree that generates predictions via a series of if-then decisions. For regression problems, CART training begins by selecting, from all available predictors, the predictor and corresponding cut point that optimally separates the criterion variable distribution into two groups. This creates a decision node with two branches. For each branch, the process is repeated, creating a new decision node. The same predictor may be selected again at a new node

⁷ Ensemble methods like bagging comprise a family of procedures that have recently received a great deal of attention in machine learning. *Boosting* is another popular form of ensemble. For more information, see Breiman (1996a), Freund & Schapire (1995), Kearns & Valiant (1989), and Valiant (1984).

with a different cut point, or a different predictor may be selected instead. Each branch eventually terminates, once a stopping rule is met, with a predicted value for the criterion variable. In regression problems, the predicted value of a CART model is equal to the average value of all cases in the training data set that meet the branch's decision criteria. The fitting of a CART model often concludes with a "pruning" process in which the model is simplified by trimming back decision branches.

CART models have several attractive features; for example, they are computationally efficient, easy to interpret, deal well with missing data, perform automatic feature selection, account for variable interactions, and make no distributional assumptions about the data. Unfortunately, the flexibility of CART models comes at a cost; they are highly prone to overfitting, as minor perturbations in the training data may substantially influence the form of the final model (Breiman, 1996b; Hastie et al, 2009; James et al., 2014). Thus, CART models tend to generalize poorly. Put in terms of the bias-variance trade-off, CART models have low bias but high variance. However, the low bias, high flexibility, and computational efficiency of CART models makes them excellent candidates for bagging, which compensates for the high variance of individual CART models by aggregating their predictions. The random forests algorithm is essentially a bagged CART ensemble augmented with additional features to improve predictive accuracy⁸. Random forests are among the most powerful and generalizable prediction methods currently in use in machine learning (Caruana & Niculescu-Mizil, 2006; Efron & Hastie, 2016; Fernández-Delgado, Cernadas, Barro, & Amorim, 2014).

⁸ Perhaps most notably, only a prespecified number of predictors are considered at each decision point in each CART model in the ensemble. Use of such a tuning variable tends to improve the ensemble's overall predictive accuracy by decorrelating the individual trees from one another and thereby increasing the diversity of their predictions.

Support vector regression. Support vector machines (SVMs; Smola & Vapnik, 1997) were originally designed to handle classification problems (for more information on the support vector machine for classification, see the discussion under “Classification Methods,” below). A technical explication of the extension of SVMs to regression problems, known unsurprisingly as support vector regression (SVR; Gunn, 1998; Smola & Schölkopf, 2004) is beyond the scope of this paper. For now, suffice to say that despite the complexity of the underlying mechanics, SVR is fairly simple for a user to apply in practice (Dimitriadou, Hornik, Leisch, Meyer, & Weingessel, 2005) and tends to produce good predictive performance (Drucker et al., 1997; ; Mattera & Haykin, 1999; Müller et al., 1997; Stitson et al., 1999; Were, Bui, Dick, & Singh, 2014). Among its many potential advantages, SVR is capable of accounting for nonlinearities in the data without substantially increasing the complexity of the predictive model. Furthermore, an SVR model can be penalized in a manner analogous to regularization in order to reduce unwanted model variance. In the present study, optimization of SVR tuning parameters (e.g., variables controlling model penalty and flexibility around nonlinearity) are performed via automated algorithms (Dimitriadou, Hornik, Leisch, Meyer, & Weingessel, 2005).

Classification methods.

Maximum likelihood estimation logistic regression. Logistic regression is a popular generalization of linear regression that extends the linear prediction procedure to the task of classification. Rather than producing a point prediction along a continuous criterion variable for a particular case, a standard logistic regression formula provides an estimate of the probability that a particular case takes membership in one of two exhaustive, non-overlapping classes. Logistic regression weights are typically estimated via maximum likelihood, a generalization of least squares. Thus, with regard to producing generalizable predictions, logistic regression

inherits limitations of OLS linear regression (for example, that increased model complexity tends to increase model variability, leading to unstable parameter estimates). Logistic regression is widely used in both psychology research and machine learning.

The prediction formula for logistic regression can be written as

$$\widehat{Pr}(X) = \frac{e^{\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p}}{1 + e^{\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p}}, \quad (7)$$

where $\widehat{Pr}(X)$ is the predicted probability that a given observation takes membership in a target discrete, dichotomous class (adopted from James et al., 2014). From this estimated probability, class membership is predicted. Typically, if $\widehat{Pr}(X) \geq 0.5$, then the observation is predicted to take membership in the target class; otherwise, not.

Ridge logistic regression. Ridge regression (introduced under “Ridge Linear Regression,” above) functions similarly for logistic regression as it does for linear regression. During parameter estimation, the model is penalized for estimation of large predictor weights. As in linear regression, this penalty results in more conservative predictor weight estimates, potentially improving generalizability by introducing a modest amount of model bias to constrain excessive model variance.

Random forests classification. The random forests (Breiman, 2001) algorithm operates very similarly for classification problems as it does for regression problems (see “Random Forest Regression,” above). For a dichotomous outcome, at each decision node, a predictor and cut point are selected that separate the target class and complement class most efficiently. Each branch in the tree eventually terminates in a yes/no decision indicating predicted membership or non-membership, respectively, in the target class. The predicted value for each terminal node represents the most common group status for individuals in the training data who meet that branch’s decision criteria. Thus, each classification tree in a random forests ensemble offers a

dichotomous prediction for any new observation. These results are then aggregated into a predicted probability of membership in the target class, which is in turn converted into a dichotomous prediction of group membership status.

Support vector machines. Support vector machines (SVMs; Smola & Vapnik, 1997) are predictive models that conduct classification by defining a boundary through predictor space to maximally separate the classes. In the linear case, this boundary is a hyperplane: a flat plane that passes through all dimensions of the data. However, SVM boundaries need not be linear; in fact, an SVM can accommodate non-linearity in a computationally efficient manner by incorporating non-linear functions known as *kernels* in the boundary fitting. One of the most distinctive features of SVM modeling is the way in which the decision boundary (the imaginary plane through the data that maximally separates one class from the other) is defined. During model fitting, only observations occurring near the decision boundary (in a region within a certain distance from the boundary known as *soft margin*) and those violating the margin (i.e., training cases falling on the “wrong” side of the margin) are used to define the boundary. These defining data points are known as the *support vectors*. Any other cases, which includes a large portion of many training data sets, are essentially ignored and do not directly define the decision boundary of the final model. Put more simply, the decision boundary of an SVM model is mostly controlled by a particularly contentious (in terms of classification) subset of training cases. Fitting of the SVM boundary is partly controlled using a tuning variable C , which sets a penalty for boundary violations. The value of C is typically chosen via cross-validation; excessively high values result in over-fitting, whereas excessively low values lead to under-fitting (Hastie et al., 2009; James et al., 2014).

Though difficult to interpret, particularly if non-linear kernels are employed, SVMs are nevertheless powerful and relatively easy to implement (Dimitriadou, Hornik, Leisch, Meyer, & Weingessel, 2005; Efron & Hastie, 2016; James et al., 2014), performing competitively in external validation with other advanced prediction models, such as random forests (Caruana & Niculescu-Mizil, 2006; Fernández-Delgado, Cernadas, Barro, & Amorim, 2014; Statnikov, Wang, & Alferis, 2008). SVMs' success may be due at least in part to the way in which the decision boundary is defined; by defining the boundary purely in terms of support vectors, the SVM is relatively robust to the influence of outliers.

Feature selection methods.

Single-variable selection. Given the premium placed on model parsimony in the psychological sciences, researchers often seek to make formal predictions based on the fewest explanatory variables possible. This is perhaps especially the case in the domain of psychological assessment, where psychologists often work with multi-scale instruments, yet tend to focus on the predictive validity of individual scales and indices. And although psychologists performing assessments using multi-scale instruments typically do informally integrate information from multiple scales to generate predictions, the functioning of such measures is generally described using bivariate statistics (e.g., zero-order correlations, performance metrics for single-scale cut scores). Furthermore, the psychological assessment literature is rife with studies identifying the “best” predictors of various criteria from among larger pools of potential predictors, often leading to the implicit suggestion or explicit recommendation that the identified “best” indicator be used in relative isolation as a predictor of the criterion. To evaluate the generalizability of applied predictions generated using such a process, the following standardized procedure is adopted for the present study.

First, zero-order associations (Pearson correlations for regression criteria, AUC statistics for classification criteria) are calculated between the prediction criteria and all potential predictors (i.e., all MMPI-2-RF substantive scales) in the training data. Then, a bivariate prediction model is created for each prediction criterion. For each such model, the potential predictor evidencing the strongest zero-order association serves as the model's sole predictive feature. The zero-order associations between the regression prediction criteria (RRF scales) and the potential predictors (MMPI-2-RF substantive scales) are listed in Table 5. Similarly, AUC statistics describing the associations between the classification prediction criteria (diagnostic group memberships) and the MMPI-2-RF substantive scales are listed in Table 6. The composition of each single-variable model is described in Table 7.

Table 5

Zero-Order Correlations Between RRF Scales and MMPI-2-RF Substantive Scales

	County Hospital				VA Medical Center			
	Depressed	Substance Abuse	Delusions	Pain	Depressed	Substance Abuse	Delusions	Pain
EID	.41	.10	-.30	.05	.27	.14	-.19	.06
BXD	.08	.51	-.09	.02	.03	.51	-.01	.05
THD	-.13	.02	.27	.03	-.06	.04	.24	.02
RCd	.41	.12	-.31	.04	.29	.16	-.18	.08
RC1	.18	.01	-.16	.15	.08	-.06	-.08	.22
RC2	.37	.02	-.25	.05	.20	.04	-.14	.05
RC3	.05	.03	-.04	.05	-.02	.06	.01	.03
RC4	.17	.53	-.13	.03	.10	.57	.00	.03
RC6	-.07	.02	.27	.03	-.06	.05	.29	.01
RC7	.23	.10	-.11	.02	.11	.15	-.07	.06
RC8	-.04	.07	.11	.00	.00	.09	.09	.02
RC9	-.03	.22	.00	-.01	-.02	.25	.00	.06
MLS	.39	.07	-.28	.15	.19	.02	-.18	.17
GIC	.19	.03	-.15	.07	.13	-.04	-.10	.06
HPC	.15	-.02	-.18	.25	.08	-.02	-.06	.25
NUC	.03	.04	-.05	.08	.00	-.03	-.02	.15
COG	.20	.08	-.15	.05	.15	.02	-.11	.15
SUI	.36	.05	-.32	.00	.22	.16	-.16	.02
HLP	.31	.01	-.24	.03	.16	-.01	-.16	.07
SFD	.38	.09	-.32	.03	.30	.12	-.18	.06

	County Hospital				VA Medical Center			
	Depressed	Substance Abuse	Delusions	Pain	Depressed	Substance Abuse	Delusions	Pain
NFC	.23	.08	-.16	.02	.17	.06	-.11	.08
STW	.29	.09	-.21	.06	.21	.09	-.11	.09
AXY	.22	.07	-.12	.03	.09	.11	-.07	.07
ANP	.14	.13	-.09	.04	.08	.15	-.09	.08
BRF	.05	-.05	.02	-.01	-.06	-.11	.01	.02
MSF	.05	-.07	.00	-.01	-.06	-.08	.01	.03
JCP	.07	.38	-.06	.03	.03	.38	.00	.05
SUB	.19	.58	-.12	-.02	.10	.61	-.02	-.02
AGG	.10	.24	-.12	.03	.05	.32	-.06	.08
ACT	.03	.13	.02	-.01	.05	.18	-.03	.04
FML	.09	.09	-.02	.04	.09	.19	.06	-.02
IPP	.20	-.03	-.14	.01	.10	-.03	-.06	-.05
SAV	.20	.00	-.12	.03	.08	-.02	-.09	.01
SHY	.21	.04	-.09	.01	.07	.05	-.07	-.01
DSF	.06	.02	-.03	.01	-.06	.05	-.02	.01
AES	-.10	-.08	.16	-.04	-.07	.08	.10	-.02
MEC	-.10	.15	.04	.05	-.01	.06	.02	.03
AGGR-r	-.19	.10	.09	.01	-.11	.12	.04	.08
PSYC-r	-.07	.03	.23	.01	-.05	.04	.21	.01
DISC-r	.06	.49	-.06	.01	.03	.50	.01	.03
NEGE-r	.29	.12	-.19	.06	.17	.12	-.11	.08
INTR-r	.27	-.04	-.19	.05	.12	-.05	-.09	.02

Note. EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 6

AUCs between Diagnostic Groups and MMPI-2-RF Substantive Scales

	County Hospital				VA Medical Center				Community Mental Health			
	Dep. Dx	SUD Dx.	Thought Dx	Som. Dx	Dep. Dx	SUD Dx.	Thought Dx	Som. Dx	Dep. Dx	SUD Dx.	Thought Dx	Som. Dx
EID	.74	.56	.71	.52	.65	.56	.60	.47	.69	.51	.63	.58
BXD	.52	.73	.60	.80	.56	.74	.51	.62	.52	.75	.48	.65
THD	.56	.51	.63	.69	.56	.51	.70	.49	.54	.54	.70	.61
RCd	.73	.57	.69	.50	.64	.56	.58	.50	.69	.53	.59	.47
RC1	.63	.53	.59	.68	.60	.50	.59	.74	.65	.52	.56	.73
RC2	.74	.52	.65	.58	.67	.51	.58	.55	.66	.50	.59	.56
RC3	.53	.51	.58	.69	.48	.53	.51	.48	.58	.56	.48	.58
RC4	.58	.75	.63	.72	.55	.77	.51	.59	.54	.76	.49	.67
RC6	.55	.51	.60	.72	.46	.51	.67	.47	.53	.52	.64	.62
RC7	.63	.54	.61	.45	.55	.56	.50	.46	.63	.52	.49	.58
RC8	.49	.54	.58	.63	.47	.54	.61	.54	.57	.57	.71	.56
RC9	.55	.59	.54	.81	.59	.62	.49	.49	.48	.62	.45	.63
MLS	.71	.56	.68	.77	.64	.51	.62	.67	.66	.50	.48	.72
GIC	.61	.51	.60	.49	.59	.52	.60	.62	.59	.50	.56	.55
HPC	.61	.53	.41	.56	.58	.51	.58	.68	.62	.51	.60	.65
NUC	.54	.54	.50	.66	.55	.50	.53	.73	.61	.56	.58	.80
COG	.63	.55	.58	.46	.60	.50	.54	.54	.64	.53	.53	.72
SUI	.72	.55	.68	.53	.60	.58	.43	.49	.64	.55	.52	.72
HLP	.69	.51	.65	.55	.62	.48	.45	.47	.61	.48	.55	.50
SFD	.71	.56	.69	.59	.63	.53	.57	.52	.66	.53	.53	.51
NFC	.63	.53	.61	.56	.58	.52	.51	.54	.61	.51	.55	.49
STW	.64	.53	.67	.45	.61	.54	.45	.51	.59	.52	.48	.58
AXY	.62	.53	.41	.61	.56	.56	.52	.46	.61	.53	.54	.52
ANP	.56	.55	.60	.52	.51	.58	.44	.50	.58	.55	.71	.46
BRF	.52	.48	.49	.50	.52	.47	.56	.60	.55	.51	.68	.62
MSF	.52	.47	.49	.60	.51	.46	.55	.55	.55	.57	.65	.78
JCP	.54	.65	.44	.79	.47	.67	.51	.63	.51	.64	.56	.57
SUB	.56	.79	.62	.64	.54	.81	.50	.57	.52	.85	.48	.42

	County Hospital				VA Medical Center				Community Mental Health			
	Dep. Dx	SUD Dx	Thought Dx	Som. Dx	Dep. Dx	SUD Dx	Thought Dx	Som. Dx	Dep. Dx	SUD Dx	Thought Dx	Som. Dx
AGG	.56	.60	.60	.59	.49	.65	.45	.57	.56	.65	.65	.62
ACT	.45	.53	.49	.65	.42	.58	.52	.51	.49	.54	.55	.68
FML	.56	.52	.58	.61	.47	.59	.55	.52	.61	.51	.54	.56
IPP	.65	.48	.46	.72	.61	.48	.50	.64	.56	.55	.48	.71
SAV	.66	.52	.57	.63	.61	.49	.52	.42	.59	.48	.44	.56
SHY	.64	.54	.43	.59	.61	.51	.52	.64	.60	.49	.55	.63
DSF	.56	.51	.50	.42	.50	.52	.56	.47	.55	.52	.50	.52
AES	.56	.55	.59	.61	.57	.53	.53	.57	.49	.51	.63	.62
MEC	.57	.59	.51	.88	.48	.55	.50	.59	.43	.60	.64	.55
AGGR-r	.63	.55	.52	.76	.61	.56	.48	.59	.54	.60	.54	.61
PSYC-r	.53	.51	.61	.66	.45	.50	.68	.49	.56	.53	.71	.60
DISC-r	.51	.72	.57	.85	.57	.73	.52	.67	.54	.71	.51	.70
NEGE-r	.64	.55	.67	.59	.58	.56	.55	.50	.62	.52	.61	.50
INTR-r	.71	.50	.60	.61	.65	.52	.56	.45	.64	.47	.56	.58

Note. AUC = Area Under the Curve; Dep. Dx = Depressive Disorder Diagnosis; SUD Dx = Substance Use Disorder Diagnosis; Thought Dx = Thought Disorder Diagnosis; Som. Dx = Somatic Disorder Diagnosis; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 7

Single-Predictor Models

Regression			Classification		
Criterion	Predictor	Pearson's <i>r</i>	Criterion	Predictor	AUC
Depressed	EID	.41	Depressive Dx	RC2	.74
Substance Abuse	SUB	.58	SUD Dx	SUB	.79
Delusions	SFD	-.32	Thought Disorder Dx	EID	.71
Pain	HPC	.25	Somatic Disorder Dx	MEC	.88

Note. AUC = Area Under the Curve; Dep. Dx = Depressive Disorder Diagnosis; SUD Dx = Substance Use Disorder Diagnosis; Thought Dx = Thought Disorder Diagnosis; Som. Dx = Somatic Disorder Diagnosis; EID = Emotion/Internalizing Dysfunction; HPC = Head Pain Complaints; MEC = Mechanical-Physical Interests; RC2 = Low Positive Emotionality; SFD = Self-Doubt; SUB = Substance Abuse.

Theory-driven (manual) feature selection. When developing models in psychology and other applied scientific disciplines, researchers often advocate for theory-driven feature selection. That is, rather than including all possible predictors or allowing an automated algorithm to select predictors based on some statistical criterion of model fit, predictors are carefully selected by the researcher based on the theoretical strength of association between the potential predictor and the criterion. To empirically compare this approach to automated feature selection (as well as full feature inclusion), the author has pre-selected sets of predictor variables, based on theoretical considerations and with an eye toward parsimony, to fit conventional linear or logistic regression models to the criterion variables. The selected predictors, along with predictor scale descriptions adapted from descriptive material in the MMPI-2-RF Interpretive manual (Ben-Porath & Tellegen, 2008), are listed in Table 8.

Table 8

Manually Selected Predictor Variables for Theory-Driven Models

Criterion	Predictor	Predictor Description
Depressed (RRF); Depressive Disorder	RCd (Demoralization)	Unhappiness, dissatisfaction with life circumstances
	RC2 (Low Positive Emotions)	Lack of positive emotional experiences
	MLS (Malaise)	Sense of poor health and general physical unwellness
Substance Abuse (RRF); Substance Use Disorder	SUB (Substance Abuse)	Past or current substance abuse, primarily alcohol abuse
	DISC-r (Disconstraint-revised)	Poor impulse control, acting out, sensation and excitement seeking
Delusions; Thought Disorder	RC6 (Ideas of Persecution)	Uncommon persecutory beliefs
	RC8 (Aberrant Experiences)	Unusual thoughts and perceptual experiences
Pain (RRF)	MLS (Malaise)	General sense of poor health and physical unwellness
	GIC (Gastrointestinal Complaints)	Gastrointestinal problems, e.g., poor appetite, nausea, vomiting
	HPC (Head Pain Complaints)	Head and neck pain
Somatic Disorder	RC1 (Somatic Complaints)	A range of perceived physical and cognitive problems
	MEC (Mechanical-Physical Interests) (-)	Interest in mechanical and physical pursuits, e.g., fixing or building things, sports

Note. RRF = Record Review Form; (-) = negative expected association with criterion variable.

Stepwise selection procedures. In psychology research, the term “stepwise selection” is often used to refer to a family of related procedures: forward selection, backward deletion, and forward-and-backward stepwise selection. Though features can be selected for nearly any type of model using stepwise-type methods, these procedures are most often applied in a regression framework (as they are in the present study). In forward selection, predictors are incrementally introduced into the model based upon whether they add meaningfully to prediction, according to some model evaluation criterion. Backward deletion begins with the full model, using all predictors, then systematically removes variables that do not sufficiently contribute to prediction. Forward-and-backward stepwise selection begins similarly to forward selection, but at each subsequent stage may either add or remove a variable until the algorithm determines that neither adding nor removing a variable will substantially improve prediction.

In the present study, models constructed via stepwise selection are evaluated using the Akaike information criterion (AIC; Akaike, 1973). AIC is a penalized badness-of-fit index designed to estimate error of prediction based on training data performance while accounting for model complexity. Thus, the optimal model produced by each stepwise selection procedure is the one that minimizes AIC within the constraints of the procedure’s search parameters.

Lasso. The lasso (least absolute shrinkage and selection operator; Tibshirani, 1996), a regularization method similar to ridge regularization, is notable for its ability to simultaneously perform shrinkage and feature selection (i.e., selective inclusion of predictors). Like ridge regression, the lasso aims to minimize an optimization function that penalizes large parameter estimates (James et al., 2014; notation simplified here):

$$RSS + \lambda \sum |\beta_p|. \quad (1)$$

At a glance, the lasso optimization penalty appears very similar to the ridge penalty. However, the ridge regression optimizer shrinks all beta weight estimates *toward* zero (while still assigning each predictor a nonzero weight), but when λ is sufficiently large, the lasso shrinks some beta weights until they become *exactly* equal to zero, effectively removing them from the final prediction equation. The lasso's regularization tends to lead to stable parameter estimates, similar to ridge regression, and the lasso's feature selection tends to produce relatively parsimonious prediction models. And like ridge regression, the lasso has been extended to a number of prediction problems beyond linear regression, including classification via logistic regression (Tibshirani, 1996).

Results

As a reminder, the data and validation procedures used in the present study are described using machine learning terminology. “Training data” refers to the dataset used to calibrate models: in this case, the county hospital inpatient mental health dataset. “Cross-validation” here refers specifically to k -fold cross validation (i.e., validation conducted using the k -fold cross-validation resampling procedure on the training data, as described previously). “External validation” refers to validation on data drawn from settings other than that of the training data. In the case of the regression models described here, the inpatient VAMC dataset is used for all external validation. For the classification models described later, “external validation” refers to validation in the VAMC or community outpatient samples, as specified.

Regression Models

Regression models predicting record review form (RRF) criterion scales are presented in Tables 9-12, separated by prediction criterion. Unstandardized predictor beta weights are presented for all applicable models. Random forests algorithms do not use beta weights; thus, as an alternative, variable importance values are presented for the predictors used in the random forests ensembles described in Tables 9-12⁹. No variable importance metrics were available for

⁹ The presented variable importance metrics for random forests regression ensembles are *node purity* metrics. Higher node purity values indicate greater predictor importance. These values represent the extent to which residual sum of squares (RSS) is reduced by splitting along the indicated variable, across the many decision trees comprising the ensemble. Note that highly correlated predictors will “share” node purity between them, resulting in lower node

the support vector regression models used in the present study; as such, only model performance summary statistics are presented for such models.

Two types of model performance statistics are presented for regression models: R^2 and root mean square error (RMSE). As noted previously, R^2 is most appropriately applied to linear models; for that reason, R^2 values are presented here only for prediction models that are linear in form. R^2 is a “goodness-of-fit” statistic; a higher value indicates a better fit of the model to the data. It is typically interpreted as the proportion of variance in the data accounted for by the model, with potential values ranging from 0 to 1. Adjusted R^2 , a modified form of the training data R^2 that is increasingly penalized (shrunk) as the number of predictors in the model increases, is also presented where appropriate in order to provide an additional model comparison metric that may be familiar to psychology researchers. The penalty term in the adjusted R^2 statistic is designed to compensate for overfitting in the model to the training data due to inclusion of multiple predictors; thus, adjusted R^2 provides an additional estimate of generalizable model performance, albeit one that may not predict generalizability as well as some other penalized metrics (see James et al., 2014). In contrast to R^2 , RMSE is a “badness-of-fit” statistic; the higher the value, the greater the error of prediction. It is interpretable as the average error of prediction in the scale of the unstandardized prediction criterion, and the statistic can take any nonnegative value.

Model performance statistics are presented for three sets of data. First, training data performance is presented. As discussed above, training data performance is often a poor indicator of model generalizability due to the potential for overfitting; nevertheless, it is presented here because model fit in psychology research is usually evaluated based on model

purity values for each variable than would be observed if one of the intercorrelated predictor variables was omitted from the training data.

performance in the training data. Also included here is k -fold cross-validated model performance ($k = 10$ in most cases; see discussion of k -fold cross-validation under “Data Analysis Strategy,” above). Although the present study focuses primarily on external validation, k -fold cross-validation is frequently used in machine learning to estimate generalizability when external validation data are not available. Finally, external validation performance statistics are presented. These values were calculated by applying the models calibrated in the training dataset to an entirely new dataset gathered in a new setting (in this case, an inpatient psychiatric ward at a VA medical center). External validation performance statistics for all regression models are consolidated in Table 13.

Table 9

Regression Models Predicting RRF Depression

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	1.09	.12	1.06			1.06	1.09			.69	.65	.95	.96	.96	1.03
EID						.01	.01	159		.06		.01			
BXD						-.03	-.01	62							
THD						-.20	-.05	76				-.19	-.19	-.19	-.06
RCd	.09	.05	.08	381		.04	.02	148			.05	.05	.05	.05	.03
RC1	.01	.01	.01	245		.03	.01	70							
RC2	.02	.01	.03	326		-.03	.01	118			.03				
RC3	-.01	-.01	-.01	203		.00	.00	66							
RC4	.02	.01	.02	241		.01	.01	68							
RC6	-.03	-.02	-.03	191		.06	.00	56				.05	.05	.05	
RC7	.00	.00	.00	258		-.01	.00	65							
RC8	-.08	-.04	-.07	201		.00	-.02	66							
RC9	-.02	-.01	-.01	229		-.03	-.01	70				-.04	-.04	-.04	-.01
MLS						.14	.10	118			.11	.13	.14	.14	.13
GIC						-.02	.01	39							
HPC						-.06	-.01	40							
NUC						-.10	-.06	52				-.06	-.06	-.06	-.05
COG						-.07	-.03	45				-.06	-.06	-.06	-.03
SUI						.13	.12	158				.15	.16	.16	.14
HLP						.05	.03	52							.02
SFD						.04	.06	70							.04
NFC						-.05	-.02	55				-.05	-.05	-.05	-.01
STW						-.07	.00	53							
AXY						.04	.04	35							.01
ANP						-.04	-.02	40							
BRF						-.02	-.01	37							
MSF						.02	.01	51							
JCP						-.01	-.01	43							
SUB						.07	.06	60				.08	.08	.08	.06

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
AGG						-.04	-.03	44				-.06	-.06	-.06	-.03
ACT						.10	.05	45				.09	.09	.09	.03
FML						-.02	-.02	61							-.01
IPP						-.02	-.01	53							
SAV						-.03	-.01	51							
SHY						.01	.01	48							
DSF						-.13	-.10	35				-.13	-.12	-.12	-.10
AES						-.03	-.02	51							-.01
MEC						.02	.01	49							
AGGR-r						-.02	-.01	65							-.01
PSYC-r						.11	.01	57				.11	.11	.11	
DISC-r						.02	.00	63							
NEGE-r						.08	.03	77				.04	.04	.04	.02
INTR-r						.03	.00	70							
R^2_{train}	.21		.21			.28	.27			.17	.19	.27	.27	.27	.26
$Adj. R^2_{train}$	.20					.25				.17	.18				
R^2_{CV}	.21		.20			.22	.23			.17	.19	.24	.24	.24	.24
R^2_{EV-VA}	.10		.10			.11	.12			.07	.08	.11	.11	.11	.12
$RMSE_{train}$	1.48	1.50	1.48	.86	1.02	1.41	1.43	.64	1.39	1.52	1.50	1.42	1.42	1.42	1.43
$RMSE_{CV}$	1.49	1.51	1.49	1.51	1.59	1.48	1.46	1.47	1.50	1.52	1.51	1.46	1.46	1.46	1.46
$RMSE_{EV-VA}$	1.53	1.52	1.52	1.54	1.38	1.54	1.49	1.53	1.45	1.56	1.59	1.49	1.48	1.48	1.49

Note. RRF = record review form; *RMSE* = root mean square error; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); OLS = ordinary least squares; NB = negative binomial; RF = random forests; SVR = support vector regression; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 10

Regression Models Predicting RRF Substance Abuse

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	.62	-.12	.64			1.11	.90			.82	.30	1.05	.86	1.05	.81
EID						.02	.00	84							
BXD						.18	.06	321				.19	.17	.19	.04
THD						.05	.01	57							
RCd	.00	.00	.00	345		.01	.01	73							
RC1	-.01	.00	-.01	351		.01	.00	83							
RC2	-.03	-.01	-.02	314		-.04	-.02	72					-.04		
RC3	-.04	-.02	-.03	318		-.02	-.02	87					-.02		-.02
RC4	.27	.13	.25	1212		.07	.07	416				.06	.07	.06	.06
RC6	.00	.00	.00	240		.01	.01	53							
RC7	-.03	-.01	-.02	345		-.02	-.01	78							
RC8	-.01	.00	-.01	269		-.02	-.01	58							
RC9	.00	.00	.01	422		-.02	-.01	104				-.04		-.04	
MLS						-.01	.00	48							
GIC						.03	.03	51							
HPC						-.03	-.03	46							
NUC						-.03	-.01	59							
COG						.02	.01	57							
SUI						-.10	-.08	49				-.10	-.10	-.10	-.08
HLP						-.06	-.05	40				-.06		-.06	-.03
SFD						-.07	-.03	28							
NFC						-.02	-.02	56							
STW						.03	.02	53							
AXY						.10	.08	34				.09	.10	.09	
ANP						.00	.01	59							
BRF						.00	-.01	43							
MSF						-.06	-.03	64				-.07	-.06	-.07	-.02
JCP						.01	.04	105							.04
SUB						.42	.35	751		.59	.46	.43	.42	.43	.41

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
AGG						-.10	-.04	65				-.12	-.13	-.12	
ACT						.04	.03	69				.07		.07	
FML						-.01	-.01	62							
IPP						.02	.02	71					.04		
SAV						.05	.03	70					.04		
SHY						-.07	-.05	66				-.06	-.07	-.06	-.03
DSF						.06	.04	44							
AES						-.07	-.07	83				-.07	-.06	-.07	-.04
MEC						.04	.01	57				.05	.05	.05	
AGGR-r						-.01	-.01	84							
PSYC-r						-.03	-.01	58							
DISC-r						-.10	.01	228			.10	-.10	-.10	-.10	
NEGE-r						-.01	.00	69							
INTR-r						-.03	-.01	80							
R^2_{train}	.31		.31			.42	.41			.34	.36	.41	.41	.41	.40
$Adj. R^2_{train}$	.30					.39				.34	.36				
R^2_{CV}	.30		.31			.38	.37			.35	.37	.39	.38	.38	.39
R^2_{EV-VA}	.32		.32			.38	.39			.37	.39	.38	.38	.38	.40
$RMSE_{train}$	1.69	1.75	1.69	.86	1.27	1.55	1.56	.69	1.48	1.65	1.62	1.56	1.56	1.56	1.58
$RMSE_{CV}$	1.70	1.77	1.71	1.74	1.90	1.61	1.61	1.62	1.64	1.64	1.62	1.60	1.60	1.61	1.60
$RMSE_{EV-VA}$	1.51	1.58	1.50	1.55	1.70	1.46	1.43	1.45	1.47	1.45	1.43	1.45	1.46	1.45	1.42

Note. RRF = record review form; *RMSE* = root mean square error; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); OLS = ordinary least squares; NB = negative binomial; RF = random forests; SVR = support vector regression; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 11

Regression Models Predicting RRF Delusions

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	1.16	.29	1.17			1.06	1.02			1.21	.45	1.03	.92	.88	.93
EID						-.04	-.01	54				-.02	-.04	-.04	
BXD						-.01	.00	25							
THD						.08	.04	80				.05	.04	.05	.05
RCd	-.06	-.10	-.05	195		.00	-.01	80				-.01			-.01
RC1	-.02	-.04	-.02	105		.01	.00	28							
RC2	.00	-.01	.00	121		.05	.00	29					.04	.04	
RC3	-.02	-.03	-.02	114		-.02	-.01	38				-.02	-.02	-.02	
RC4	-.01	-.01	-.01	116		-.03	-.01	31					-.04		
RC6	.12	.13	.12	178		.07	.06	83			.12	.09	.08	.09	.07
RC7	.03	.06	.02	114		.03	.01	26				.03	.02	.02	
RC8	.04	.09	.04	112		-.01	.00	29			-.01				
RC9	-.01	-.02	-.01	116		-.02	.00	24							
MLS						-.04	-.03	32					-.04	-.03	-.03
GIC						.01	.01	10							
HPC						-.07	-.05	20				-.07	-.06	-.06	-.05
NUC						-.01	.00	19							
COG						-.01	-.01	19							
SUI						-.09	-.08	78				-.10	-.10	-.10	-.10
HLP						-.02	-.03	14							-.01
SFD						-.06	-.07	41		-.23		-.07	-.06	-.06	-.08
NFC						.01	.01	25							
STW						.01	-.01	21							
AXY						-.03	-.01	10							
ANP						.03	.02	20						.03	
BRF						.03	.02	16							.01
MSF						-.01	-.01	20							
JCP						.04	.01	18					.05		
SUB						.04	.01	17					.04		

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
AGG						-.02	-.02	17						-.04	-.01
ACT						.01	.00	21							
FML						.02	.02	26					.03		
IPP						-.01	.00	19							
SAV						.02	.01	23				.02			
SHY						.04	.02	20					.04	.04	.01
DSF						.05	.04	19				.05	.05	.06	.03
AES						.06	.05	35				.06	.05	.06	.04
MEC						-.01	.00	26							
AGGR-r						.00	.00	27				-.02			
PSYC-r						-.02	.02	46							
DISC-r						.02	.00	28							
NEGE-r						-.01	.00	31							
INTR-r						-.01	.00	31							
R^2_{train}	.23		.23			.28	.27			.10	.07	.27	.28	.28	.27
$Adj. R^2_{train}$	.22					.25				.10	.07				
R^2_{CV}	.22		.22			.23	.24			.11	.08	.24	.23	.23	.25
R^2_{EV-VA}	.13		.13			.14	.15			.03	.08	.14	.14	.14	.15
$RMSE_{train}$	1.00	1.05	1.00	.52	.43	.96	.97	.43	.89	1.08	1.09	.97	.97	.97	.97
$RMSE_{CV}$	1.01	1.06	1.01	1.02	1.11	1.00	.99	.99	1.02	1.07	1.10	1.00	1.00	1.00	.99
$RMSE_{EV-VA}$	.87	.92	.87	.90	.92	.87	.85	.88	.84	.92	.89	.88	.87	.87	.86

Note. RRF = record review form; *RMSE* = root mean square error; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); OLS = ordinary least squares; NB = negative binomial; RF = random forests; SVR = support vector regression; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 12

Regression Models Predicting RRF Pain

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	.14	-1.97	.15			.04	.09			.08	.05	.05	.06	.05	.07
EID						.01	.00	11							
BXD						.02	.00	10							
THD						.06	.01	9					.05		
RCd	.00	.00	.00	40		-.01	.00	10				-.01	-.01	-.01	
RC1	.03	.11	.02	49		-.03	.00	12				-.02	-.02	-.02	-.01
RC2	.00	-.01	.00	41		-.03	.00	10							
RC3	.01	.04	.01	37		.01	.01	9					.01		
RC4	.01	.04	.00	41		.00	.00	10							
RC6	.01	.05	.01	33		-.01	.00	8							
RC7	-.01	-.04	-.01	39		-.01	.00	9				.00			
RC8	-.01	-.06	-.01	33		-.01	-.01	8							
RC9	-.01	-.03	-.01	44		-.01	-.01	11					-.01		
MLS						.07	.03	11			.02	.06	.06	.07	.04
GIC						.00	-.01	7			-.04				-.01
HPC						.16	.09	15		.10	.10	.15	.15	.15	.12
NUC						.02	.00	9							
COG						.00	.00	8							
SUI						-.02	-.02	7				-.03	-.03	-.02	-.02
HLP						.01	.00	6							
SFD						-.01	-.01	6							
NFC						.00	.00	9							
STW						.00	.01	7							
AXY						-.01	-.01	6							-.01
ANP						-.01	.00	7							
BRF						-.01	-.01	6							-.01
MSF						-.02	-.01	9				-.01	-.02	-.02	-.01
JCP						.00	.01	8							
SUB						-.02	-.01	8							

	RC Scales					All Substantive Scales				Feature Selection					
	OLS	NB	Ridge	RF	SVR	OLS	Ridge	RF	SVR	Single	Manual	Forward	Backward	Stepwise	Lasso
AGG						.00	.00	8							
ACT						.00	-.01	8							
FML						.00	.00	10							
IPP						.00	.00	9							
SAV						-.01	.00	9							
SHY						.00	.00	8							
DSF						.00	.00	6							
AES						.00	.00	8							
MEC						.04	.02	9				.03	.03	.03	.02
AGGR-r						-.01	.00	9							
PSYC-r						-.04	-.01	8					-.05		
DISC-r						-.02	.00	10							
NEGE-r						.01	.00	10							
INTR-r						.01	.00	9							
R^2_{train}	.04		.03			.12	.11			.06	.07	.10	.11	.10	.10
$Adj. R^2_{train}$	.03					.08				.06	.07				
R^2_{CV}	.04		.04			.06	.06			.08	.08	.08	.07	.08	.08
R^2_{EV-VA}	.04		.04			.06	.07			.06	.07	.06	.06	.06	.07
$RMSE_{train}$	.69	.69	.69	.42	.51	.66	.66	.41	.57	.68	.68	.66	.66	.66	.66
$RMSE_{CV}$	.69	.69	.69	.70	.70	.68	.68	.69	.71	.67	.68	.67	.67	.67	.68
$RMSE_{EV-VA}$	.75	.76	.76	.76	.79	.75	.74	.75	.78	.75	.74	.75	.75	.75	.74

Note. RRF = record review form; *RMSE* = root mean square error; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); OLS = ordinary least squares; NB = negative binomial; RF = random forests; SVR = support vector regression; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 13

External Validation (VA Medical Center) Performance of Regression Models

Model	R^2				RMSE			
	Depressed	Sub. Abuse	Delusions	Pain	Depressed	Sub. Abuse	Delusions	Pain
<i>RC Scales</i>								
Ordinary Least Squares	.10	.32	.13	.04	1.53	1.51	.87	.75
Negative Binomial					1.52	1.58	.92	.76
Ridge	.10	.32	.13	.04	1.52	1.50	.87	.76
Random Forests					1.54	1.55	.90	.76
Support Vector Regression					1.38	1.70	.92	.79
<i>All Substantive Scales</i>								
Ordinary Least Squares	.11	.38	.14	.06	1.54	1.46	.87	.75
Ridge	.12	.39	.15	.07	1.49	1.43	.85	.74
Random Forests					1.53	1.45	.88	.75
Support Vector Regression					1.45	1.47	.84	.78
<i>Feature Selection</i>								
Single Predictor	.07	.37	.03	.06	1.56	1.45	.92	.75
Manual Selection	.08	.39	.08	.07	1.59	1.43	.89	.74
Forward Selection	.11	.38	.14	.06	1.49	1.45	.88	.75
Backward Deletion	.11	.38	.14	.06	1.48	1.46	.87	.75
Stepwise Selection	.11	.38	.14	.06	1.48	1.45	.87	.75
Lasso	.12	.40	.15	.07	1.49	1.42	.86	.74

A comparison of regression model performances in the training data with cross-validated and externally validated performances reveals a clear and expectable trend. That is, models generally fit training data the best, fit k -fold cross-validation data somewhat less well, and fit external validation data least well. However, this general trend did not hold true for all regression prediction criteria. Perhaps most notably, models predicting the Substance Abuse record review form scale tend to fit the external validation data approximately as well as, or sometimes even better than, the training data, as reflected in both R^2 and RMSE values. The cause of this unexpected finding is not obvious, though possible contributing factors include sampling variability and unusually strong model generalizability. In further support of the latter hypothesis, models predicting the Substance Abuse record review form scale tend to show stronger predictive validity in both training and test data than models predicting other regression criteria, as evidenced by higher R^2 values for the models predicting Substance Abuse. In general, for models for which significant discrepancies between training and external validation data performance are observed, cross-validation performance appears to fall much closer to training data performance than to external validation performance. As notable exceptions to this rule, the random forests and support vector regression models appear to overfit the training data to a much greater extent than any other modeling approach considered, though they nevertheless appear to produce external validation performance roughly comparable to other approaches. Models predicting the Pain record review form scale generally evidenced substantially poorer fit to the external validation data than models predicting other record review form scales. This likely reflects the overall low rate and variability of pain in both the training and external validation data.

R^2 and RMSE values for the models displayed in Tables 9-12 appear to generally agree with one another when used to compare models predicting the same regression criterion. For example, a low R^2 for one model, relative to the R^2 value of a competing model predicting the same criterion (indicating worse fit to the data), typically corresponds to a higher RMSE (also indicating worse fit). The models predicting the Delusions record review form scale are a notable exception to this trend. Models predicting Delusions tend to evidence substantially lower R^2 values in the external validation data than in the training data, indicating a loss of generalizable predictive validity. However, the same models also tend to show a decrease in RMSE relative to training data performance when applied to external validation data, indicating an *increase* in predictive validity. This seemingly paradoxical finding can potentially be explained by the fact that the mean Delusions score in the external validation sample is substantially lower than the mean Delusions score in the training sample. Similarly, variability of the Delusions variable is lower in the external validation sample than in the training sample (see Table 3). As such, the discrepancy between externally validated R^2 and RMSE values in the Delusions models could reflect a range restriction problem to which R^2 , a measure of multiple correlation, is more sensitive than RMSE, a measure of absolute fit.

A full discussion of each of the vast number of possible comparisons that can be made between the different regression models created as part of the present study is beyond the scope of this paper. Briefly, models predicting each of the regression criteria did not perform identically when applied to the external validation data, though models predicting the same criterion did appear to produce external validation performances in the same general range. More specifically, external validation RMSEs ranged from 1.38-1.59 for Depression, 1.42-1.70 for Substance Abuse, .86-.92 for Delusions, and .74-.79 for Pain.

The specific modeling approach producing the lowest RMSE varied between regression criteria. Support Vector Regression (SVR) using the RC Scales as predictors produced the lowest RMSE (1.38) for the Depression criteria, relative to other models. Curiously, SVR models produced external validation error rates among the *highest* of all competing methods when predicting Substance Abuse, Delusions, and Pain regression criteria (RMSE = 1.70, .92, and .79, respectively). The lasso produced the lowest RMSE among all models predicting Substance Abuse (1.42). The SVR model predicting Delusions using all Substantive Scales produced the lowest external validation RMSE (.84) among all competing models, though the other Delusions SVR model (i.e., the one using only the RC Scales as predictors) produced the highest external validation RMSE, as already noted. Several models were tied for the overall lowest external validation RMSE (.74) in predicting Pain (perhaps unsurprising given the narrow range of external validation performances for this particular criterion): ridge regression using all Substantive Scales, the manually selected model, and the lasso model.

To summarize the relative performances of each modeling approach both within and across regression criteria, a simple rank sum score was calculated for each approach. This score was computed by ranking each model within each regression criterion by external validation RMSE (lowest to highest, using the rounded values displayed in Tables 9-13), then summing each model's ranks across all four regression criteria. Thus, the lower the score, the better the performance of the modeling approach's performance across regression criteria, relative to other modeling approaches, as measured by external validation RMSE. The results of this coarse post-hoc analysis are displayed in Table 14. Rank sum scores for regression modeling approaches range from 10 (lasso regression and ridge regression using the full Substantive Scale set as predictors) to 48 (random forests regression using only the RC scales as predictors). The two

highest-performing regression modeling approaches are regularized forms of regression (the lasso and the ridge). Four of the five top-performing methods are automated feature selection methods (the lasso, forward-and-backward stepwise regression, backward deletion, and forward selection, in that order). Modeling using all Substantive Scales as predictors generally performed better than models using only the RC Scales as predictors. Models constructed using manual (theory-based) feature selection or using a single, automatically selected predictor were systematically outperformed by automated feature selection methods permitted to consider and include multiple predictors.

Table 14

Rank Sum Scores for External Validation Performance of Regression Models Based on RMSE

Modeling Approach	Feature Set	Rank Sum Score
Lasso	Feature Selection	10
Ridge	All Substantive Scales	10
Stepwise Selection	Feature Selection	15
Backward Deletion	Feature Selection	19
Forward Selection	Feature Selection	22
Random Forests	All Substantive Scales	27
Support Vector Regression	All Substantive Scales	27
Ordinary Least Squares	All Substantive Scales	28
Manual Selection	Feature Selection	29
Ordinary Least Squares	RC Scales	30
Ridge	RC Scales	34
Single Predictor	Feature Selection	35
Support Vector Regression	RC Scales	44
Negative Binomial Regression	RC Scales	46
Random Forests	RC Scales	48

Classification Models

Classification models predicting diagnostic group membership are presented in Tables 15-18. As with the regression models already discussed, unstandardized beta weights are presented for all available model. Because random forests classifiers do not use beta weights,

variable importance metrics are presented instead¹⁰. No variable importance metrics were available for support vector machines; thus, only overall model performance statistics are included for those models.

Performance for classification models is described using two types of statistic: overall model accuracy and area under the curve (AUC). Importantly, these metrics evaluate different aspects of model performance. Overall classification accuracy, also known as hit rate, is simply the proportion of correct classifications (hits) to all classifications. As such, it is a measure of the model's ability to correctly make a dichotomous (in this case) classification decision. On the other hand, AUC reflects a trade-off between sensitivity and specificity across the full range of a continuous model output variable, in this case the predicted class probabilities. AUC values typically range from .5 (chance prediction) to 1.0 (perfect prediction), though values below .5 are also possible and indicate a model that generally performs worse than chance. Of note, in machine learning, classification accuracy is generally preferred over AUC as a model performance measure when class base rates are approximately equal; however, AUC is increasingly preferred over accuracy as base rates become increasingly imbalanced.

As with the regression models described above, model performance metrics for training data, k -fold cross-validated test data¹¹, and external validation data are presented for the classification models generated as part of the present study. Two external validation samples were available for the present study: the inpatient VA medical center sample also used to validate the regression models described previously, and an outpatient community mental health

¹⁰ The variable importance metric used for random forests classifiers in the present study, Gini decrease in impurity, is interpreted similarly to the node purity metric used for random forests regression ensembles. That is, a higher value indicates greater variable importance, though multicollinearity between predictors can attenuate estimates of variable importance.

¹¹ Note that k -fold cross-validated classification accuracy was not computed in the present study for random forests or support vector machines due to software limitations.

care center sample. Thus, external validation model performance statistics are presented for both samples. External validation performance is across all classifications in summarized in Table 19 for classification accuracy and in Table 20 for AUC.

Similar to the regression models already described, performance for the classification models used in the present study almost always appears strongest in the training data, next strongest in the *k*-fold cross-validation data, and least strong in the external validation data, regardless of the model performance criterion under consideration. An exception to this is observed when using classification accuracy to evaluate the performance of models with low base rates in the external validation sample (e.g., thought disorder diagnosis in the community mental health sample). In these cases, classification accuracy tends to appear much higher in the external validation sample than in the training sample, likely due to the increasing influence of base rates on classification accuracy as base rates approach their extremes. As an additional exception to the general performance rule (training performance > cross-validated performance > externally validated performance), some models predicting substance use disorders produce higher performance in the external validation data than in the training data, mirroring similar findings in predicting the substance abuse record review form variable using regression models. Classification accuracy and AUC otherwise appear to generally agree on model performance within classification criterion (e.g., relatively high performance, as measured by classification accuracy, appears to be relatively high AUC). Nevertheless, this seems not to be a hard and fast rule, and a model that appears to perform slightly better than another according to one performance metric does not necessarily outperform the same model according to the other performance metric. For example, in predicting a substance use disorder diagnosis, the lasso logistic model appears to outperform the maximum likelihood estimation logistic model (when

using all substantive scales as predictors) according to overall classification accuracy in the community mental health sample (lasso accuracy = .81, MLE accuracy = .79). However, this comparative observation is reversed when using AUC as the performance criterion (lasso AUC = .85, MLE AUC = .86).

External validation performance of classification models appears to vary considerably between some prediction criteria, though various models predicting the same criterion variables tend to produce performance values in the same range within each setting, similar to findings from the regression model analyses (depressive disorder accuracy range = .57-.65 [VA], .52-.66 [community]; substance use disorder accuracy range = .65-.76 [VA], .69-.81 [community]; thought disorder accuracy range = .84-.86 [VA], .97-.98 [community] somatic disorder range = .97-.98 [VA], .98-1.00 [community]; depressive disorder AUC range = .64-.71 [VA], .66-.70 [community]; substance use disorder AUC range = .69-.82 [VA], .73-.86 [community]; thought disorder AUC range = .58-.75 [VA], .59-.76 [community]; somatic disorder AUC range = .49-.70 [VA], .49-.72 [somatic]). Notably, some models produce very different external validation performances between settings, particularly when base rates differ markedly, and even when using AUC as the performance criterion (e.g., full scaleset support vector machine VAMC AUC = .50, community mental health center AUC = .58). However, this appears not to be the case for most modeling approaches. The highest-performing classification method for each diagnostic variable will not be listed here, as such a determination would vary depending on both the performance metric used and the external validation setting; instead, the reader is referred to Tables 19 and 20.

Table 15

Classification Models Predicting Depressive Disorder Diagnosis

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	-1.12	-1.15			-1.00	-1.18			-1.52	-1.93	-1.17	-1.33	-1.17	-1.14
EID					.06	.01	40				.05	.05	.05	.04
BXD					-.11	-.01	11					-.06		
THD					-.07	-.03	13				-.07	-.07	-.07	-.06
RCd	.10	.07	90		.00	.02	33			.04				
RC1	.03	.03	57		.03	.01	14							
RC2	.07	.08	76		-.04	.02	24		.11	.12				
RC3	.03	.02	47		.01	.00	12							
RC4	.03	.02	51		.06	.01	11					.06		
RC6	-.12	-.11	41		-.10	-.05	11				-.09	-.08	-.09	-.06
RC7	.01	.02	56		.02	.01	13							
RC8	-.08	-.07	41		-.02	-.02	10							
RC9	-.07	-.05	60		-.01	-.01	19				-.04		-.04	-.02
MLS					.12	.07	15			.09	.12	.10	.12	.11
GIC					-.01	.03	7							
HPC					.05	.05	7				.09	.10	.09	.05
NUC					-.04	-.01	7							
COG					-.03	.00	8							
SUI					.24	.14	31				.28	.26	.28	.23
HLP					.04	.05	10							
SFD					.05	.07	12							.06
NFC					-.04	.00	9							
STW					-.06	.01	8							
AXY					-.01	.01	5							
ANP					-.09	-.03	9					-.07		-.01
BRF					-.03	-.02	6							
MSF					.01	-.01	10							
JCP					.05	.01	7							
SUB					-.04	.00	7							
AGG					.06	.01	8							

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
ACT					-.12	-.05	13				-.11	-.14	-.11	-.08
FML					-.01	.00	11							
IPP					-.01	.02	10							.01
SAV					-.01	.01	10							
SHY					.06	.04	10				.07	.06	.07	.04
DSF					-.09	-.03	5							
AES					-.04	-.02	10							-.01
MEC					.01	.00	9							
AGGR-r					-.03	-.01	13							-.01
PSYC-r					.03	-.02	11							
DISC-r					.04	.00	11							
NEGE-r					.02	.01	14							
INTR-r					.02	.02	18							
Acc _{train}	.72	.71	1.00	.89	.73	.73	1.00	.77	.69	.69	.73	.73	.73	.73
Acc _{CV}	.71	.71			.69	.72			.69	.69	.72	.70	.72	.72
Acc _{EV-VA}	.63	.64	.63	.57	.64	.65	.63	.65	.59	.62	.65	.65	.65	.64
Acc _{EV-CMH}	.59	.60	.59	.52	.64	.64	.62	.64	.59	.61	.66	.65	.66	.66
AUC _{train}	.78	.78	1.00	.95	.81	.80	1.00	.84	.73	.75	.80	.80	.80	.80
AUC _{CV}	.77	.70	.76	.72	.77	.78	.79	.79	.73	.77	.78	.78	.78	.78
AUC _{EV-VA}	.70	.70	.69	.67	.72	.72	.70	.72	.64	.68	.71	.71	.71	.71
AUC _{EV-CMH}	.69	.69	.66	.66	.70	.69	.68	.69	.69	.69	.70	.70	.70	.70

Note. Acc = classification accuracy; AUC = Area Under the Curve; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); EV-CMH = external validation data (community mental health sample); MLE = maximum likelihood estimator; RF = random forests; SVM = support vector machine; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRP = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 16

Classification Models Predicting Substance Use Disorder Diagnosis

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	-1.60	-1.54			-1.73	-1.53			-2.12	-1.94	-2.11	-1.84	-2.11	-1.47
EID					.01	.00	12							
BXD					.14	.04	28					.12		
THD					.12	.01	8							
RCd	.03	.01	52		.06	.02	10				.05	.05	.05	
RC1	.02	.01	53		.01	.01	12							
RC2	-.06	-.03	50		-.09	-.03	10							
RC3	-.03	-.03	49		-.02	-.02	12							-.01
RC4	.26	.21	128		.19	.09	39				.20	.20	.20	.09
RC6	-.01	-.02	38		-.01	.00	8							
RC7	-.06	-.04	51		-.02	-.01	11							
RC8	.01	.00	38		-.01	.00	9							
RC9	-.02	.00	57		.00	-.01	13							
MLS					.05	.04	9							
GIC					-.08	-.06	6							
HPC					.10	.07	7				.11	.12	.11	.04
NUC					.01	.03	9							
COG					.04	.02	9							
SUI					-.03	-.02	6							
HLP					-.15	-.10	7				-.18	-.17	-.18	-.07
SFD					-.01	.03	4							
NFC					.00	-.01	8							
STW					-.18	-.08	9				-.15	-.14	-.15	-.04
AXY					.03	.03	5							
ANP					-.05	-.02	9							
BRF					.01	-.02	7							
MSF					-.07	-.02	9					-.06		
JCP					-.17	-.03	11				-.15	-.14	-.15	
SUB					.44	.37	81		.22	.54	.41	.44	.41	.45
AGG					-.20	-.06	9				-.16	-.22	-.16	

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
ACT					-.04	-.03	9							
FML					-.08	-.04	12				-.09	-.10	-.09	-.03
IPP					-.01	-.02	10							-.01
SAV					.05	.03	9							
SHY					.00	.00	10							
DSF					.04	.01	6							
AES					-.10	-.09	12				-.12	-.10	-.12	-.07
MEC					.10	.05	12				.07	.11	.07	.04
AGGR-r					.04	.02	12				.06	.06	.06	
PSYC-r					-.10	-.02	9							
DISC-r					-.13	.01	19			.06		-.15		
NEGE-r					.04	.00	10							
INTR-r					.01	.00	11							
Acc _{train}	.70	.70	1.00	.93	.76	.75	1.00	.77	.70	.73	.75	.76	.75	.74
Acc _{CV}	.69	.69			.73	.74			.70	.73	.73	.74	.74	.61
Acc _{EV-VA}	.71	.71	.70	.65	.75	.75	.76	.75	.71	.74	.75	.75	.75	.76
Acc _{EV-CMH}	.73	.74	.69	.74	.79	.79	.73	.77	.70	.78	.79	.79	.79	.81
AUC _{train}	.77	.77	1.00	.96	.83	.82	1.00	.85	.75	.80	.82	.82	.82	.81
AUC _{CV}	.76	.77	.72	.69	.76	.80	.78	.79	.75	.76	.80	.79	.80	.80
AUC _{EV-VA}	.77	.77	.74	.69	.80	.81	.80	.81	.77	.82	.81	.81	.81	.82
AUC _{EV-CMH}	.77	.77	.75	.73	.86	.84	.83	.82	.76	.84	.84	.86	.84	.85

Note. Acc = classification accuracy; AUC = Area Under the Curve; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); EV-CMH = external validation data (community mental health sample); MLE = maximum likelihood estimator; RF = random forests; SVM = support vector machine; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRP = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 17

Classification Models Predicting Thought Disorder Diagnosis

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	-.34	-.35			-1.20	-.64			-.65	-2.14	-.91	-.79	-1.12	-.71
EID					-.17	-.02	9		-.09		-.20	-.20	-.20	-.07
BXD					-.17	-.01	7							
THD					.03	.06	10				.24		.24	.05
RCd	-.12	-.09	35		.08	-.01	8				.08	.08	.09	
RC1	-.05	-.04	26		-.04	-.01	6							
RC2	.02	-.01	29		.22	.01	7				.21	.24	.23	.09
RC3	-.09	-.08	26		-.09	-.04	7				-.07	-.08	-.08	-.07
RC4	-.08	-.07	29		-.12	-.03	7					-.18		-.05
RC6	.18	.18	31		.19	.08	8			.15		.21		.15
RC7	.01	-.01	28		.07	.00	7					.07		
RC8	.23	.20	29		.14	.07	8			.03		.16		.13
RC9	-.02	-.02	29		-.03	-.01	7							
MLS					-.13	-.07	7				-.14	-.14	-.15	-.09
GIC					.01	-.03	4							
HPC					-.01	-.04	4							-.04
NUC					.09	.04	5							.01
COG					.00	.01	6							
SUI					-.16	-.11	6				-.15	-.17	-.15	-.14
HLP					-.11	-.08	5							-.07
SFD					-.08	-.09	6							-.04
NFC					.05	.02	6							.03
STW					.02	-.07	7							-.05
AXY					-.10	-.03	4							-.01
ANP					.09	-.01	5							
BRF					-.02	.00	4							
MSF					.04	-.01	6							
JCP					.14	.03	5					.18		.02
SUB					-.09	-.09	5				-.15		-.16	-.11
AGG					.08	-.02	5							

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
ACT					.02	.00	6							
FML					.00	-.02	6							-.01
IPP					.05	.03	6							.02
SAV					-.03	-.01	6							
SHY					.15	.04	5				.18	.17	.19	.09
DSF					.12	.07	4							.08
AES					.11	.08	6				.12	.12	.12	.10
MEC					-.14	-.03	6					-.10		-.05
AGGR-r					.03	-.02	7				-.03			-.02
PSYC-r					.00	.06	9							
DISC-r					.24	.02	6					.08		.03
NEGE-r					-.11	-.03	7					-.08		-.05
INTR-r					.00	.01	7							
Acc _{train}	.84	.84	.99	.96	.85	.85	1.00	.98	.83	.83	.85	.85	.85	.85
Acc _{CV}	.83	.84			.83	.84			.83	.83	.84	.83	.84	.84
Acc _{EV-VA}	.85	.86	.85	.86	.86	.86	.86	.84	.86	.86	.86	.86	.86	.86
Acc _{EV-CMH}	.98	.98	.98	.98	.97	.98	.98	.98	.98	.99	.97	.97	.97	.98
AUC _{train}	.81	.81	1.00	.99	.85	.83	1.00	1.00	.69	.61	.83	.84	.83	.84
AUC _{CV}	.80	.75	.78	.70	.80	.81	.79	.75	.69	.80	.79	.79	.79	.81
AUC _{EV-VA}	.74	.75	.72	.67	.74	.76	.71	.70	.58	.67	.76	.75	.75	.76
AUC _{EV-CMH}	.75	.75	.75	.76	.72	.76	.71	.58	.59	.67	.77	.74	.77	.75

Note. Acc = classification accuracy; AUC = Area Under the Curve; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); EV-CMH = external validation data (community mental health sample); MLE = maximum likelihood estimator; RF = random forests; SVM = support vector machine; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRP = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 18

Classification Models Predicting Somatic Disorder Diagnosis

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
Intercept	-3.18	-4.09			-131.5	-4.49			-3.31	-3.97	-2825.6	-455.32	-2825.6	-4.64
EID					-4.31	.00	.41							
BXD					3.92	-.02	.34					173.07		
THD					3.08	-.03	.30							
RCd	.08	.01	2.25		3.59	.00	.44							
RC1	.12	.07	2.06		-5.81	.02	.40			.05				
RC2	-.10	.01	1.27		-5.17	.00	.32							
RC3	-.07	-.05	1.45		.01	-.03	.35							-.01
RC4	-.12	-.06	1.23		-5.34	-.02	.21					-171.31		
RC6	-.49	-.12	.54		-14.50	-.05	.15				-268.24	-235.70	-268.24	-.11
RC7	.07	.00	1.13		-4.56	-.01	.30							
RC8	-.17	-.06	.72		-5.45	-.02	.10							
RC9	-.16	-.07	1.45		-2.35	-.02	.40					-66.97		
MLS					13.36	.10	.34				259.68	137.75	259.68	.30
GIC					9.18	.03	.46							
HPC					-2.90	-.02	.16							
NUC					6.48	.06	.38							
COG					5.60	.00	.22							
SUI					13.55	.02	.28				267.73	133.06	267.73	
HLP					-3.03	.00	.14							
SFD					-15.11	-.07	.26				-359.04		-359.04	-.15
NFC					-.52	-.02	.27							
STW					-7.51	.03	.25							
AXY					-3.85	.06	.18							
ANP					6.66	.01	.39				156.33		156.33	
BRF					-7.99	-.05	.12				-262.06	-215.98	-262.06	
MSF					2.80	.05	.47				133.31		133.31	
JCP					-8.77	-.09	.16				-471.57	-227.47	-471.57	-.28
SUB					-.71	-.02	.35							
AGG					.32	.00	.15							

	RC Scales				All Substantive Scales				Feature Selection					
	MLE	Ridge	RF	SVM	MLE	Ridge	RF	SVM	Single	Manual	Forward	Backward	Stepwise	Lasso
ACT					-3.73	-.04	.33							
FML					8.61	.06	.27					104.91		.02
IPP					7.28	.05	.43				171.60		171.60	.04
SAV					11.29	.01	.21					101.17		
SHY					.29	-.05	.32				-69.46		-69.46	-.01
DSF					-8.41	-.07	.07							
AES					4.03	.07	.41							.04
MEC					-10.17	-.11	.25		-.21	-1.96	-503.72	-421.77	-503.72	-.40
AGGR-r					2.06	-.03	.24							
PSYC-r					4.22	-.02	.21							
DISC-r					2.11	-.04	.31							-.04
NEGE-r					12.25	.02	.28				42.56	59.93	42.56	
INTR-r					-3.16	.00	.33					-102.46		
Acc _{train}	.99	.99	1.00	1.00	1.00	.99	1.00	.99	.99	.99	1.00	1.00	1.00	.99
Acc _{CV}	.99	.99			.98	.99			.99	.99	.98	.98	.99	.99
Acc _{EV-VA}	.98	.98	.98	.98	.97	.98	.98	.98	.98	.98	.97	.98	.97	.98
Acc _{EV-CMH}	1.00	1.00	1.00	.99	.98	1.00	1.00	1.00	1.00	1.00	.98	.98	.98	1.00
AUC _{train}	.91	.90	1.00	1.00	1.00	.99	1.00	1.00	.81	.91	1.00	1.00	1.00	.98
AUC _{CV}	.70	.62	.96	.50	.70	.84	.88	.79	.81	.70	.57	.49	.70	.84
AUC _{EV-VA}	.61	.62	.56	.64	.51	.67	.68	.66	.49	.61	.50	.50	.50	.70
AUC _{EV-CMH}	.62	.72	.61	.56	.60	.72	.72	.74	.63	.49	.49	.49	.49	.64

Note. Acc = classification accuracy; AUC = Area Under the Curve; train = training data (county hospital sample); CV = *k*-fold cross-validation; EV-VA = external validation data (VAMC sample); EV-CMH = external validation data (community mental health sample); MLE = maximum likelihood estimator; RF = random forests; SVM = support vector machine; EID = Emotional/Internalizing Dysfunction; BXD = Behavioral/ Externalizing Dysfunction; THD = Thought Dysfunction; RCd = Demoralization; RC1 = Somatic Complaints; RC2 = Low Positive Emotions; RC3 = Cynicism; RC4 = Antisocial Behavior; RC6 = Persecutory Ideation; RC7 = Dysfunctional Negative Emotions; RC8 = Aberrant Experiences; RC9 = Hypomanic Activation; MLS = Malaise; GIC = Gastrointestinal Complaints; HPC = Head Pain Complaints; NUC = Neurological Complaints; COG = Cognitive Complaints; SUI = Suicidal/Death Ideation; HLP = Helplessness/Hopelessness; SFD = Self-Doubt; NFC = Inefficacy; STW = Stress/Worry; AXY = Anxiety; ANP = Anger Proneness; BRF = Behavior-Restricting Fears; MSF = Multiple Specific Fears; JCP = Juvenile Conduct Problems; SUB = Substance Abuse; AGG = Aggression; ACT = Activation; FML = Family Problems; IPP = Interpersonal Passivity; SAV = Social Avoidance; SHY = Shyness; DSF = Disaffiliativeness; AES = Aesthetic-Literary Interests; MEC = Mechanical-Physical Interests; AGGR-r = Aggressiveness-Revised; PSYC-r = Psychoticism-Revised; DISC-r = Disconstraint-Revised; NEGE-r = Negative Emotionality-Revised; INTR-r = Introversion-Revised.

Table 19

External Validation Performance of Classification Models (Overall Classification Accuracy)

Model	VA Medical Center				Community Outpatient			
	Depressive	Substance	Thought	Somatic	Depressive	Substance	Thought	Somatic
<i>RC Scales</i>								
Maximum Likelihood	.63	.71	.85	.98	.59	.73	.98	1.00
Ridge	.64	.71	.86	.98	.60	.74	.98	1.00
Random Forests	.63	.70	.85	.98	.59	.69	.98	1.00
Support Vector Machine	.57	.65	.86	.98	.52	.74	.98	.99
<i>All Substantive Scales</i>								
Maximum Likelihood	.64	.75	.86	.97	.64	.79	.97	.98
Ridge	.65	.75	.86	.98	.64	.79	.98	1.00
Random Forests	.63	.76	.86	.98	.62	.73	.98	1.00
Support Vector Machine	.65	.75	.84	.98	.64	.77	.98	1.00
<i>Feature Selection</i>								
Single Predictor	.59	.71	.86	.98	.59	.70	.98	1.00
Manual Selection	.62	.74	.86	.98	.61	.78	.99	1.00
Forward Selection	.65	.75	.86	.97	.66	.79	.97	.98
Backward Deletion	.65	.75	.86	.98	.65	.79	.97	.98
Stepwise Selection	.65	.75	.86	.97	.66	.79	.97	.98
Lasso	.64	.76	.86	.98	.66	.81	.98	1.00

Note. Depressive = Depressive Disorders; Substance = Substance Use Disorders; Thought = Thought Disorders; Somatic = Somatic Disorders.

Table 20

External Validation Performance of Classification Models (Area Under the Curve)

Model	VA Medical Center				Community Outpatient			
	Depressive	Substance	Thought	Somatic	Depressive	Substance	Thought	Somatic
<i>RC Scales</i>								
Maximum Likelihood	.70	.77	.74	.61	.69	.77	.75	.62
Ridge	.70	.77	.75	.62	.69	.77	.75	.72
Random Forests	.69	.74	.72	.56	.66	.75	.75	.61
Support Vector Machine	.67	.69	.67	.64	.66	.73	.76	.56
<i>All Substantive Scales</i>								
Maximum Likelihood	.72	.80	.74	.51	.70	.86	.72	.60
Ridge	.72	.81	.76	.67	.69	.84	.76	.72
Random Forests	.70	.80	.71	.68	.68	.83	.71	.72
Support Vector Machine	.72	.81	.70	.66	.69	.82	.58	.74
<i>Feature Selection</i>								
Single Predictor	.64	.77	.58	.49	.69	.76	.59	.63
Manual Selection	.68	.82	.67	.61	.69	.84	.67	.49
Forward Selection	.71	.81	.76	.50	.70	.84	.77	.49
Backward Deletion	.71	.81	.75	.50	.70	.86	.74	.49
Stepwise Selection	.71	.81	.75	.50	.70	.84	.77	.49
Lasso	.71	.82	.76	.70	.70	.85	.75	.64

Note. Depressive = Depressive Disorders; Substance = Substance Use Disorders; Thought = Thought Disorders; Somatic = Somatic Disorders.

Prediction of somatic disorder diagnosis in the present data appears particularly problematic, relative to prediction of other classification criteria. The somatic disorder diagnostic variable differs from other diagnostic variables in the present study in its low base rate across all samples (see Table 3¹²), an issue that typically limits predictive power at both training and testing stages. Indeed, many of the models predicting somatic disorder classification appear to predict no better than chance, as reflected in overall classification accuracy less than or equal to the base rate of the dominant class in the sample and in AUC values no greater than approximately .50). Furthermore, overall predictive performance of somatic disorder classifiers appears relatively low compared to classifiers for other diagnostic groups. In addition, the predictor weights calibrated by some approaches (namely, forward selection, backward deletion, and forward-and-backward stepwise selection) are wildly out of bounds compared to the weights calibrated other logistic regression approaches, indicating extreme overfitting to the training data. Nevertheless, several modeling approaches appear to substantially improve upon chance-level prediction of somatic disorder classification (e.g., lasso AUC = .70 [VA], .64 [community]).

As a post-hoc analysis, rank sum scores were calculated for the classification models produced in the present study. This process was similar to the one used to calculate rank sum scores for the regression models, with some notable exceptions. First, only AUC performance was considered, as classification accuracy proved a problematic metric for evaluating performance in low base rate settings, as previously described. Second, model performance in predicting somatic disorder diagnosis was not considered because of the already-discussed problems in training and testing those models resulting from low statistical power. Third, performance was considered across both external validation settings. Thus, six ranks were

¹² The low base rate of clinically diagnosable somatization in these samples is not surprising, as individuals with such problems tend to present at medical settings, rather than mental health care settings, for treatment.

calculated for each modeling approach: one for each of the three remaining classification criteria (depressive disorder diagnosis, substance use disorder diagnosis, and thought disorder diagnosis) in both external validation settings (VAMC and community mental health center). The results of this analysis are displayed in Table 21. Overall, the findings of this rank sum score analysis appear similar to the results of the corresponding analysis conducted for regression modeling approaches. Two of the top five modeling approaches, of the fourteen total classification approaches considered, are regularization methods (the lasso and ridge regression using all Substantive Scales as predictors). Four of the top five approaches use automated feature selection of potentially multiple predictors (forward selection, the lasso, forward-and-backward stepwise selection, and backward deletion, in that order). The least effective classification methods in the present study, with respect to generalizability, appear to be support vector machines using all Substantive Scales, tied with single-predictor automated selection. In general, classification models incorporating predictive information from all Substantive Scales appear to perform better than models using only the RC Scales as predictors, though lower-performing methods using all Substantive Scales (e.g., SVMs, random forests classifiers) appear to generally perform as well as, or worse than, some methods using only the RC Scales (ridge logistic regression, MLE logistic regression) or a manually selected feature set.

Table 21

Rank Sum Scores for External Validation Performance of Classification Models Based on AUC

Modeling Approach	Feature Set	Rank Sum Score
Forward Selection	Feature Selection	14
Lasso	Feature Selection	15
Stepwise Selection	Feature Selection	17
Ridge	All Substantive Scales	18
Backward Deletion	Feature Selection	22
Maximum Likelihood	All Substantive Scales	28
Ridge	RC Scales	43
Support Vector Machines	All Substantive Scales	44
Maximum Likelihood	RC Scales	46
Manual Selection	Feature Selection	47
Random Forests	All Substantive Scales	57
Random Forests	RC Scales	64
Support Vector Machines	RC Scales	69
Single Predictor	Feature Selection	69

Discussion

Results of the present study provide insight into several issues related to the generalizability of statistical prediction in the domain of psychological assessment. In the following discussion, the author first considers the extent to which statistical prediction broadly appears to generalize across settings, as well as the extent to which the loss of generalizability appears, itself, to be predictable. This is followed by commentary on the association between model parsimony and model generalizability, including a discussion of the (perhaps counterintuitive) finding that emphasizing parsimony may actually diminish generalizability. Next, the author discusses the impact of various approaches to predictor selection on generalizability, particularly with respect to theory-driven predictor selection, automated multivariate approaches (e.g., stepwise-type methods), and empirically-driven approaches that focus on identifying and applying single predictors in isolation. This is then followed by a discussion of the relative performances of the various modeling approaches considered in the present study. Finally, the author discusses limitations of the present study and directions for future research, then summarizes the major findings and implications of the study. As before, the training data are the data collected from the inpatient county hospital setting, which were used to calibrate all models; cross-validation refers to use of the k -fold cross-validation resampling procedure on the training data; and the external validation data are the datasets

collected from the VAMC inpatient setting and community outpatient setting, which were used to evaluate the generalizable predictive validity of the models developed in the training setting.

Overall Generalizability of Statistical Prediction in Psychological Assessment

The results of the present study appear to support the longstanding concern in the psychology literature that models based on psychometric data often do not perform as expected in applied settings when that expectation is based upon a model's performance in its training setting. In most cases, the models developed as part of the present study fit the external validation data noticeably more poorly than they fit the training data. For example, many of the regression models evaluated using R^2 tended to account for only about half as much variance in the external validation sample as they did in the training sample (e.g., R^2 dropping from approximately .20 to approximately .10). None of the modeling approaches considered appeared immune to this shrinkage effect, though some modeling approaches appeared to consistently produce more generalizable predictions than others. Nevertheless, many models demonstrated non-negligible predictive power in the external validation data despite significant performance shrinkage.

The extent to which models generalized in the present study varied significantly based not only on the modeling approach considered but also on the criterion variable. Somewhat unexpectedly, models predicting criterion variables related to substance abuse (i.e., the substance abuse record review form scale, substance abuse-related clinical diagnosis) appeared particularly robust against shrinkage, often performing approximately as well in external validation samples as in the training sample (sometimes even better). The reason for this outlying finding, which violates the shrinkage trend observed with all other prediction criteria examined in this study, is not entirely clear. The presence of multiple potential predictors (e.g., BXD, RC4, SUB) with

especially strong zero-order associations to the substance abuse criteria in both training and test data raises the possibility that psychometric predictors with similarly strong empirical associations to the prediction criteria may not only produce stronger overall predictions (as one might have expected *a priori*) in external validation settings, but also enjoy increased protection against a loss of generalizability across settings. Of note, the predictor pool in the present study included a particular MMPI-2-RF scale, SUB (Substance Abuse, which also shares items with RC4 and BXD), that maps very closely onto the substance use-related prediction criteria at a conceptual level. It is therefore also possible, if not likely, that this especially close theoretical association further enhanced prediction generalizability in the present study (though as indicated by other results, discussed in greater detail below, strong theoretical association alone appears insufficient to maximize model generalizability).

K -fold cross-validation performance estimates in the present study, though generally more conservative than training data estimates of performance, nevertheless typically hewed closely to those training data estimates. In cases for which a significant drop in model performance was observed when porting models from training data to external validation data, the k -fold cross-validation estimates of model performance tended not to reflect the extent to which predictive power was attenuated across settings. In a notable exception to this pattern, some of the newer machine learning algorithms considered in the present study (namely, random forests and support vector regression/machines) evidenced extreme overfitting to the training data, predicting data points in the training sample with perfect or near-perfect accuracy. However, these models' cross-validated performance estimates fell roughly in line with those of the other methods considered, as did their external validation performance estimates. This result is neither unexpected nor problematic, as those predictive methods were explicitly designed to

predict new data points, and their training data performances are therefore not particularly meaningful. The results of the present study appear to indicate that k -fold cross-validation model performance is insufficient to predict generalizable model performance across settings, though one could reasonably infer, as is often the case in machine learning studies, that it is a good estimate of model performance within the training setting (i.e., if new observations were drawn from the same population or process that generated the training sample).

The results of the present study do not appear to suggest a simple rule for anticipating the extent to which any given model will generalize across settings. Certainly, any *a priori* assumption that a model will demonstrate predictive power in an applied setting similar to its predictive power in its training data appears unwarranted. Any general assumption that a model will have no practical predictive validity outside of its training setting also appears unjustified, given that many models in the present study appeared to have meaningful predictive power in the external validation sample, even when substantial performance shrinkage was evident. How, then, should one estimate the expected performance of a model in a new, potential applied setting? The simplest and most effective option is to test the model on known data from that setting, if possible. It generally takes significantly fewer observations to test a model than it does to appropriately train a model, making model testing potentially more economically feasible than model development in a new setting. Barring that, a clinician must calibrate their expectations more informally, considering information such as the potential loss of predictive power indicated by past research (e.g., the present study), shifts in base rates and other distributional characteristics between settings, and qualitative differences between the settings. Depending on the nature and quantity of available information, a researcher can also apply a formal corrective procedure, such as model updating to account for differences between settings, even when

lacking sufficient data to either formally test an existing model or train a new one (see “Future Directions,” below). The extent to which a loss in predictive power between settings is acceptable is ultimately a matter of professional judgment that will depend on the model and the purpose to which it is applied.

The Predictive Value of Parsimony, or Lack Thereof

The results of the present study appear to undermine conventional recommendations that psychology researchers prioritize parsimony in model construction, at least when the model’s predictive power is of primary interest. In the present study, models demonstrated a systematic tendency toward stronger predictive performance when they were permitted to incorporate more features, not fewer, at the training stage. In fact, the most parsimonious models (i.e., those employing only a single predictor or a small set of manually specified predictors) were often among the least effective, in terms of predictive power, when compared to the full set of competing models predicting the same criterion variable in the external validation sample. Furthermore, models constructed using only a circumscribed, static scaleset (the MMPI-2-RF RC Scales) were generally outperformed by models permitted to simultaneously incorporate predictive information from the full pool of potential predictors (all MMPI-2-RF substantive scales).

In the present study, models benefitted from a large training sample size (much larger than one encounters in many psychology studies), which enabled estimation of seemingly stable model parameters, even when a relatively large number of predictors were retained. Indeed, it appears that when the sample size offers sufficient statistical power in a psychology study, and when one wishes to enhance the external predictive power of the model developed, one could consider including all the potential predictors that the sample size would support. These criteria

will not always be met, of course. In other potential model development scenarios, when the number of potential predictors is relatively large given the number of training data observations (when the dataset is “wide,” in data science parlance), parsimony may be a statistical necessity, and in other cases, model developers may very reasonably wish to balance predictive power with interpretability.

Optimal Feature Selection Methods for Predictive Modeling in Psychological Assessment

Perhaps one of the most striking findings from the current study is that automated feature selection methods (the lasso and stepwise-family methods) produced some of the most consistently powerful predictive models, relative to other models considered, when validated on external data, whereas models whose features were selected based on theory were generally among the poorer-performing models. These results appear to strongly contradict usual “best practice” feature selection guidelines in psychology research, which both advocate for theory-based feature selection and condemn most automated approaches. Results of the present study also appear to devalue a certain practice in psychology, particularly common in the assessment literature: namely, the tendency to focus on identification and interpretation of applied single-predictor models (e.g., single-scale interpretation), even in the presence of other potential ancillary predictors. In other words, single-predictor models, even when empirically selected, did not appear to generalize well across settings relative to multiple-predictor models in the present study.

An inspection of the predictor weights and other variable importance metrics from the more complex models constructed in the present study provides some insight into the reason that those models tended to outperform more parsimonious and theory-based models. When selecting features based on theory, researchers tend to focus on predictors with strong, positive

associations with the target criterion. However, many of the larger and more complex models in the present study appear to have optimized their predictive power by detecting and appropriately weighting not only strong convergent features, but also negatively associated features, useful ancillary features, and suppressor variables that a human researcher would not necessarily have thought to include. Though not explicitly designed to do so, the complex and automatically selected models in the present study appear to have achieved relatively good predictive performance by learning to discriminate between some of the “institutional pseudo-taxa” (concept credited to Tellegen in Meehl, 1992; see also Grove, 1991¹³) in the sample (i.e., the latent groups representing alternative pathways into mental health treatment, such as suicidality or psychosis) and by detecting important features that a human might have overlooked. For example, prediction of delusions and thought disorders was augmented by not only considering MMPI-2-RF scales positively associated with thought dysfunction, such as RC6 (Persecutory Ideation) and RC8 (Aberrant Experiences) but also by negatively weighting features more distinctively associated with internalizing problems (e.g., depression), such as RCd (Demoralization). One might reasonably hypothesize that such modeling approaches are similarly capable of detecting and leveraging important but perhaps difficult to anticipate statistical associations between variables in other applied settings.

The stepwise-type approaches considered in the present study likely benefited from the large size of the training sample. As such, these statistically “greedy” methods, which require statistical power similar to the power needed by an OLS/MLE model incorporating *all* potential predictors (see Babyak, 2004) are unlikely to perform as well under some conditions of lower

¹³ Meehl (1992) writes: “In taxometrics one has the problem of pseudo-taxa, of data sets that may behave taxonically when examined by whatever taxometric method, but that are in some sense spurious, artifactual, not “real entities.” This concern was raised most forcibly on the local scene by Auke Tellegen and has therefore come to be designated in Minnesota circles as the *Tellegen Case*.”

statistical power. Indeed, in the current study, stepwise-type approaches failed to produce stable, generalizable parameter estimates when statistical power was limited by the low training data base rate of one of prediction criteria (somatic disorder diagnosis). Of note, however, models trained using the lasso and other regularization methods tend to require less statistical power than models parameterized using traditional OLS/MLE optimization, including stepwise-type approaches¹⁴. As such, the lasso appears to be a viable and user-friendly alternative for regression with feature selection even, perhaps, when low statistical power would not permit use of stepwise-type approaches. In the case of prediction of somatic disorder diagnosis in the present study, the lasso did indeed appear to produce a relatively stable and generalizable predictive model, whereas models produced using stepwise-type methods appeared to fare no better than chance when applied to the external validation data.

Selecting a Predictive Modeling Approach in Psychological Assessment

The results of the present study can be used to inform the selection of a predictive modeling approach when working with psychometric data. There is perhaps no conclusive method to determine the extent to which one particular method is likely to outperform another without direct performance data, given the variability in model performance for each method across prediction criteria. In the present study, the author produced rank orderings for modeling approaches by aggregating relative external validation performances across predictions for each method (see Tables 14 and 21), though this does not fully convey the magnitude of the differences in model performance (which could lead to either an over-estimate or under-estimate of the magnitude of the difference). Indeed, observed differences in absolute model fit for

¹⁴ Regularized regression uses fewer effective degrees of freedom than OLS/MLE regression because the shrinkage penalty allows the model to “reclaim” degrees of freedom from the full OLS/MLE model parameters (i.e., the predictor weights).

competing models appeared, in some cases, very small. In machine learning studies, particularly those in which multiple competing models predicting the same criterion are compared, this is not unusual, and preference is usually given to the model or modeling approach whose performance suggests the potential for greatest generalizability (as estimated from k -fold cross-validated performance or some other measure of model performance), even when its advantage over competitors appears minor. When applying comparative methods in the domain of psychology research, it is perhaps best left to the researcher (or the consumer of research) to determine whether the statistical differences observed between models are substantial enough to warrant selection of one model over another.

Similarly, because the differences in performance between some of the methods evaluated in some cases appear rather slight, in applied practice the extent to which any difference would be considered meaningful would depend not only on quantitative differences in model fit, but also on the context of prediction, which would include consideration of the nature of the prediction criterion as well as the manner in which prediction would be used to guide decision-making. For example, in predicting a high-stakes criterion such as suicide risk or sex offender recidivism, prior to making an important clinical decision or formulating a psycholegal opinion, a psychologist might favor any prediction method that offers a robust advantage over another, even if that advantage is relatively small in terms of absolute model fit. This important point is perhaps easy to overlook in the present study, given that the present study's prediction criteria are somewhat arbitrary, and the models described herein were developed for methodological analyses, not for direct applied use. Regardless, if a researcher wishes to develop a prediction model using a less robust method than available alternatives, the burden is arguably on that researcher to justify their selection of modeling approach.

Comparison of model optimization methods. Traditional ordinary least squares linear regression and maximum likelihood estimation logistic regression appeared to perform moderately well, relative to other predictive methods considered; however, they also appeared to be routinely outperformed by other methods under consideration. Regularized methods (i.e., the ridge and the lasso) were remarkable for fairly consistently producing models of comparable or greater generalizable predictive power than those produced by OLS/MLE methods. This is perhaps especially noteworthy given that such regularized models take the familiar form of linear or logistic regression models and can be interpreted similarly by anyone so inclined. The relative advantage of regularized regression over OLS/MLE regression appeared greater in the present study as the number of potential predictors increased (e.g., when using all Substantive Scales rather than simply just the RC Scales).

Some researchers have recently advocated for greater use of count regression models over linear regression models in psychological assessment research, as many of the regression criteria used in that research domain take the form of count variables, and many such criteria follow skewed, non-normal distributions in the training data that would technically violate the assumptions of conventional linear regression. Negative binomial regression was selected as a representative count regression modeling approach in the present study and was tested against linear regression methods. Perhaps surprisingly, negative binomial models tended to perform relatively poorly when compared to conventional linear models, at least when evaluated based on RMSE. The most direct potential explanation for this finding appears to be that despite taking the form of count variables, the regression criteria in the present study nevertheless fit poorly to the negative binomial distribution and are better described using linear combinations of predictors. Thus, recommendations for count regression modeling in assessment research appear

at least partly misguided, based on these findings. At the very least, these results appear to indicate a need to consider and, ideally, test alternative modeling approaches before selecting a negative binomial (or other count regression) model as a final predictive approach when working with psychometric data.

Effectiveness of advanced machine learning predictive algorithms. Unexpectedly, the random forests and support vector prediction models developed and applied in the present study demonstrated underwhelming, if somewhat uneven, generalizable predictive performance relative to many of the other approaches considered. This is perhaps especially surprising given that those methods are generally highly regarded in the machine learning literature (see, e.g., Efron & Hastie, 2016; Hastie, Tibshirani, & Friedman, 2009) and have demonstrated top-of-the-line predictive performance in past comparative studies in other substantive domains (Caruana & Niculescu-Mizil, 2006; Fernández-Delgado, Cernadas, Barro, & Amorim, 2014). It appears unlikely that the present study refutes volumes of past research supporting the effectiveness of those predictive algorithms. Thus, at least two remaining possible explanations could help account for the discrepancy between expected and observed performance for the random forests and support vector models. First, the present study strongly emphasized the off-the-shelf performance of these newer machine-learning algorithms. As such, the author made few adjustments to those algorithms' tuning variables, leaving most such parameters at their default settings (though in some cases, this meant that those parameters were set empirically using k -fold cross-validation). Thus, the relatively poor performance of random forests and support vector models may reflect an over-reliance on default tuning variable values, in which case, those methods' performance could perhaps be improved through more careful calibration of tuning parameters. An initial post-hoc review of the default values used by these algorithms does

indeed reveal some potential deficiencies in those values. For example, when passed through R's 'caret' package (Kuhn, 2008), the default number of trees for random forests models is 500, only half the number recommended even by the package's author (Kuhn & Johnson, 2013), a factor that could certainly have attenuated those models' performances. This could pose at least a minor potential problem for "off-the-shelf" use of those methods, as the skill floor for knowledgeably applying them would appear somewhat higher than for alternative modeling approaches, though not necessarily prohibitively so.

The author conducted a series of post-hoc random forests regression analyses to investigate the extent to which over-reliance on default tuning values could account for the relatively lackluster performance of random forests regression in the present study. To do so, the author generated new random forests regression ensembles for the original four regression criteria (the record review form scales) with 1,000 trees, 1,500 trees, and 2,000 trees, then externally validated those ensembles on the VAMC dataset. The results of these analyses, along with the results of the original random forests ensembles (using 500 trees each) are displayed in Table 22. Perhaps surprisingly, generalizable predictive validity appeared to improve only marginally, if at all, as number of trees increased. This indicates relatively little advantage to increasing number of trees beyond the default value in the present data for any of the current regression criteria, suggesting that similar results would likely be observed in other psychological assessment data. Although this finding may increase confidence in the default value of that the tuning variable in question, it does not appear to reflect well upon the relative utility of random forests regression in applied psychological assessment research. Further evaluation of the validity of default values and settings for other tuning variables used in the

present study is beyond the scope of this paper but would likely present a useful avenue for future research.

Table 22

External Validation RMSE of Random Forests Regression for Different Numbers of Trees

Criterion	Number of Trees			
	500 (default)	1,000	1,500	2,000
Depression	1.53	1.53	1.52	1.52
Substance Abuse	1.45	1.44	1.44	1.44
Pain	.88	.88	.88	.88
Delusions	.75	.75	.75	.75

A second possible explanation for the relatively modest performance of advanced machine learning algorithms in the present study, not mutually exclusive with the one just discussed, is that those methods are not optimal for applied predictions using psychometric data. Indeed, the post-hoc analysis just conducted would appear to corroborate this possibility, at least with respect to random forests regression. But consider also, for example, that effect sizes in psychology (operationalized most relevantly in the present study as various measures of overall model performance) tend to be rather low compared to those seen in many other fields in which machine learning is applied. It is possible that the low signal-to-noise ratio in the present data reduced the random forests and support vector algorithms' ability to learn effectively from the data to such an extent that those methods were unable to distinguish themselves from more conventional approaches. In addition, it is possible that those methods are not especially robust to cross-setting validation when the training and validation settings are not identical, meaning that they may suffer from some of the same practical limitations to generalizability affecting more conventional approaches to statistical prediction.

Limitations and Future Directions

The present study evaluated only a small fraction of the available prediction methods used in areas of applied statistical prediction. Furthermore, only a small subset of the vast number of possible psychometric predictors and prediction criteria were considered, and these were evaluated in only a few clinical settings. As such, a good deal of additional comparative research is needed to evaluate the generalizability of other predictive methods, to understand the predictive associations between other feature pools and prediction criteria, and to evaluate the generalizability of those predictions across other settings. To aid this research, it would benefit the psychology research community to identify a mutually consensual set of practically important prediction criteria that can be meaningfully evaluated between settings. This would provide a structured research framework that would not only allow a systematic evaluation of how prediction of those criteria differs across settings and predictive methods but would also ultimately bolster our ability to meaningfully predict. To maximize the utility of these predictive models, their prediction criteria should be closely tied to real-world decision making.

For cases for which a model demonstrates significant performance shrinkage across settings, or in cases for which such shrinkage is expected, the applied statistics literature offers methods for correcting for differences between settings. For example, a simple and reasonably effective approach with conventional regression-form models is to update the model intercept to reflect differences in the average value of the criterion variable between settings. In this case, the only information that must be gathered from the applied setting is the average value of the criterion. Bayesian updating, which allows for re-estimation of some or all model parameters, taking into account information from both settings, is a more advanced approach. To apply Bayesian model updating requires more detailed information (or at least informed expectations) regarding the values of any updated parameters in the applied setting. For an overview of these

and other model updating methods, see Steyerberg (2009). Like any other prediction models, these updated models can be tested empirically to evaluate the validity of the correction procedures and the extent to which they enhance model generalizability.

As previously mentioned, some of the modeling methods considered in the present study, particularly random forests and support vector regression/machines, may not have performed to their full potential due to the study's emphasis on the effectiveness of "off-the-shelf" modeling performance. As such, a logical direction for future research would be to revisit those methods and to determine the extent to which their performance could be improved through experimentation with model tuning parameters not considered in the present study. Along similar lines, alternative versions of many of these algorithms exist, some of them augmented by other statistical features. For example, regularization was considered in the present study only as it applies to linear and logistic regression; however, regularization has also been used to enhance the generalizability of other forms of statistical prediction, including support vector machines and other forms of regression (e.g., negative binomial regression). The effectiveness of these methods with respect to generalizable prediction of psychological variables has not yet been assessed.

The negative binomial models developed as part of the present study's regression analyses performed relatively poorly compared to many of the other regression models considered. However, this does not mean that other count modeling approaches (of which there are many) will necessarily perform similarly poorly (or even that the negative binomial model is entirely inappropriate for working with count variables in psychology). Future comparative studies could evaluate alternative forms of count regression that may better fit the prediction criteria, including zero-inflated and zero-truncated methods, hurdle models, and advanced

methods such as generalized Poisson regression (Consul & Famoye, 1992) or COM-Poisson regression (Sellers & Shmueli, 2010).

One noteworthy finding from the present study was that predictive generalizability appeared enhanced when the model was permitted to incorporate more, rather than less predictive information (either through automated multivariable feature selection or through inclusion of numerous potentially important predictors), assuming that the training data support a model with that level of complexity. One implication of this finding, combined with clinical observations in the field of psychological assessment that assessment findings are more valid when the clinician is able to incorporate information from multiple sources, is that statistical prediction may be enhanced when driven by multimethod assessment data. For example, rather than generating a predictive model using only data from a particular self-report test (even a broadband test such as the MMPI-2-RF), such a model could also potentially incorporate, for example, data from additional tests, demographic data, encoded behavioral observations, interview responses, and informant report data. Test data need not take the form of summed test scores; for example, some predictive models using item-level data rather than scale scores may benefit from a more granular approach to prediction while potentially reducing prediction error due to variance accounted for by items not meaningfully predictive of the criterion variable. When the number of items is very large, this approach is likely feasible only when using feature selection or dimension reduction technique.

Summary and Conclusions

The results of the present study carry significant implications for the practice of predictive modeling using psychometric data. By applying a model evaluation framework rooted in fundamental machine learning practice, the author appraised the generalizability of a range of

predictive modeling approaches, including methods used both in psychology research and in modern machine learning. Results support a longstanding concern in psychology that statistical prediction models tend to overfit their calibration data. Such models may nevertheless demonstrate acceptable predictive power in new settings, though the extent to which a loss of predictive power across settings is acceptable will depend in practice upon the context of prediction. The extent to which any given model will generalize to a new setting appears difficult to anticipate without actually testing it in the new setting, as even k -fold cross-validation, a machine learning tool designed to provide a relatively unbiased estimate of model performance in the training setting, appeared incapable of predicting significant performance shrinkage (or lack thereof) across settings in the present study.

The present study's findings appear to significantly undermine some conventional recommendations regarding predictive modeling in psychology. Perhaps most importantly, the results of this study suggest that psychological theory may be substantially over-valued as a means for selecting predictors to include in statistical models, at least when generalizable model performance is a significant concern. In fact, in the present study, models whose predictors were selected based on theory were typically among the poorest performing with respect to generalizability of prediction. Along similar lines, results of the present study also appear to refute concerns in the psychology literature that multivariate models whose predictors are selected automatically, such as by stepwise-type methods, are almost necessarily excessively overfit to their calibration data, relative to other modeling approaches. On the contrary, in the present study, models whose features were selected automatically in this fashion tended to evidence stronger generalizable predictive power than models whose features were selected in other ways (e.g., based on theory). Model parsimony also appears not to be as much a virtue in

predictive modeling as conventional wisdom in psychology research would suggest. Indeed, results of the present study appear to indicate that including as many potentially important predictors as are supported by the sample size of the training data may be an effective way to enhance the generalizable predictive power of a statistical model. Traditional ordinary least squares linear regression and maximum likelihood estimation logistic regression appear modestly robust to loss of generalizability across settings. However, models produced by regularized (penalized) regression appeared even more generalizable, and because such models can be interpreted in a similar fashion to traditional regression models, they can be considered as viable alternatives to conventional regression methods¹⁵

Although it is the position of the author, bolstered in part by the results of the present study, that psychologists would likely substantially benefit from greater use of machine learning-informed methods for developing, evaluating, and applying predictive models, such a change would constitute a significant paradigm shift in psychology research and practice (see also Yarkoni & Westfall, 2017). Components of this paradigm shift could include, but are certainly not limited to, considering a greater range of statistical modeling approaches (including some that are presently unfamiliar to most psychology researchers), shifting focus from training data performance to test data performance, relaxing or abandoning some conventional modeling principles when they are not supported by external validation data, and accepting statistical prediction as a valid basis for decision-making in applied psychology. To accomplish such a shift wholesale appears improbable, given the depth with which some countervailing methodological principles are rooted in the tradition of psychological science. As the present empirical study demonstrates, however, basic machine learning methods and principles can be

¹⁵ Implementations of regularized regression can be found in many popular statistical software packages, including SPSS.

readily applied to psychological data and do not necessarily require an excessive amount of foundational background knowledge to comprehend or require access to exclusive software or hardware to utilize¹⁶. The principle benefit of the proposed paradigm shift, even if such a shift occurs gradually, through thoughtful but tentative studies of applied predictive modeling in psychology, is the potential to substantially improve real-world decision-making using psychological data across all domains in which such data are collected.

¹⁶ All predictive modeling in the present study was carried out on a home computer using R (R Core Team, 2015), a widely-used, free, and open-source statistical software package. Extensive tutorials for using R and for developing and evaluating predictive models within R are freely available on the internet and are also found in many excellent statistics texts. The R code used to develop and evaluate the models described in this study can be found in an appendix to this paper.

References

- Ægisdóttir, S., White, M. J., Spengler, P. M., Maugherman, A. S., Anderson, L. A., Cook, R. S., & Rush, J. D. (2006). The meta-analysis of clinical judgment project: Fifty-six years of accumulated research on clinical versus statistical prediction. *The Counseling Psychologist*, 34(3), 341-382.
- Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. N. Petrov & B. F. Csaki (Eds.), *Second International Symposium on Information Theory*, (pp. 267-281). Academiai Kiado: Budapest.
- Ambler, G., Brady, A.R., Royston, P. Simplifying a prognostic model: a simulation study based on clinical data. *Stat Med* 2002; 21 (24) : 3803 - 22.
- American Psychiatric Association. (1987). *Diagnostic and Statistical Manual of Mental Health Disorders (DSM-III-R)*. American psychiatric association.
- American Psychiatric Association. (1994). *Diagnostic and statistical manual of mental disorders: DSM-IV*. American Psychiatric Association.
- Anderson, N. B. (2006). Evidence-based practice in psychology. *American Psychologist*, 61(4), 271-285.
- Babayak, M. A. (2004). What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. *Psychosomatic medicine*, 66(3), 411-421.

- Ben-Porath, Y. S. (2012a). Addressing challenges to MMPI-2-RF-based testimony: Questions and answers. *Archives of Clinical Neuropsychology*, 27(7), 691-705.
- Ben-Porath, Y. S. (2012b). *Interpreting the MMPI-2-RF*. U of Minnesota Press.
- Ben-Porath, Y. S., & Tellegen, A. (2008). *MMPI-2-RF: Manual for administration, scoring and interpretation*. University of Minnesota Press.
- Berk, R. A. (2008). *Statistical learning from a regression perspective*. Springer Science & Business Media.
- Breiman, L. (1996a). Bagging predictors. *Machine learning*, 24(2), 123-140.
- Breiman, L. (1996b). Heuristics of instability and stabilization in model selection. *The annals of statistics*, 24(6), 2350-2383.
- Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.
- Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees*. CRC press.
- Browne, M. W. (2000). Cross-validation methods. *Journal of mathematical psychology*, 44(1), 108-132.
- Busemeyer, J. R., & Wang, Y. M. (2000). Model comparisons and model selections based on generalization criterion methodology. *Journal of Mathematical Psychology*, 44(1), 171-189.
- Butcher, J. N., Dahlstrom, W. G., Graham, J. R., Tellegen, A., & Kaemmer, B. (1989). *MMPI-2: Manual for administration and scoring*.
- Caruana, R., & Niculescu-Mizil, A. (2006, June). An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning* (pp. 161-168). ACM.

- Cohen, J., Cohen, P., West, S. G., & Aiken, L. S. (2013). *Applied multiple regression/correlation analysis for the behavioral sciences*. Routledge.
- Consul, P., & Famoye, F. (1992). Generalized Poisson regression model. *Communications in Statistics-Theory and Methods*, 21(1), 89-109.
- Copas, J. B. (1983). Regression, prediction and shrinkage. *Journal of the Royal Statistical Society. Series B (Methodological)*, 311-354.
- Darlington, R. B. (1968). Multiple regression in psychological research and practice. *Psychological bulletin*, 69(3), 161.
- 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.
- Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., & Weingessel, A. (2005). Misc Functions of the Department of Statistics (e1071), TU Wien. *R package version*, 1-5.
- Drucker H., Burges C.J.C., Kaufman L., Smola A., and Vapnik V. 1997. Support vector regression machines. In: Mozer M.C., Jordan M.I., and Petsche T. (Eds.), *Advances in Neural Information Processing Systems 9*, MIT Press, Cambridge, MA, pp. 155–161.
- Efron, B., & Hastie, T. (2016). *Computer Age Statistical Inference* (Vol. 5). Cambridge University Press.
- Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, 15(1), 3133-3181.

- Freund, Y., & Schapire, R. E. (1995, March). A decision-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory* (pp. 23-37). Springer Berlin Heidelberg.
- Friedman, J., Hastie, T., & Tibshirani, R. (2009). glmnet: Lasso and elastic-net regularized generalized linear models. *R package version, 1*(4).
- Grove, W. M. (1991). Validity of taxometric inferences based on cluster analysis stopping rules. In D. Cichetti & W. M. Grove (Eds.), *Thinking clearly about psychology: Essays in honor of Paul Everett Meehl* (pp. 313–329). Minneapolis, MN: University of Minnesota Press.
- Grove, W. M., & Meehl, P. E. (1996). Comparative efficiency of informal (subjective, impressionistic) and formal (mechanical, algorithmic) prediction procedures: The clinical–statistical controversy. *Psychology, Public Policy, and Law*, 2(2), 293.
- Grove, W. M., Zald, D. H., Lebow, B. S., Snitz, B. E., & Nelson, C. (2000). Clinical versus mechanical prediction: a meta-analysis. *Psychological assessment*, 12(1), 19.
- Gunn, S. R. (1998). Support vector machines for classification and regression. *ISIS technical report, 14*.
- Harrell, F. (2015). *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis*. Springer.
- Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference and prediction* (2nd Ed.). New York: Springer.
- Hilbe, J. M. (2011). Negative binomial regression. Cambridge University Press.
- Hilbe, J. M. (2014). Modeling count data. Cambridge University Press.

- Hoerl, A. E., & Kennard, R. W. (1981). Ridge Regression—1980: Advances, Algorithms, and Applications. *American Journal of Mathematical and Management Sciences*, 1(1), 5-83.
- Hunsley, J., & Meyer, G. J. (2003). The incremental validity of psychological testing and assessment: conceptual, methodological, and statistical issues. *Psychological assessment*, 15(4), 446.
- Hunsley, J., & Mash, E. J. (2007). Evidence-based assessment. Annual Review of Clinical Psychology, 3, 29–51.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2014). *An introduction to statistical learning*. New York: Springer.
- Kearns, M., & Valiant, L. (1994). Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1), 67-95.
- Kohavi, R. (1995, August). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai* (Vol. 14, No. 2, pp. 1137-1145).
- Kotov, R., Krueger, R. F., Watson, D., Achenbach, T. M., Althoff, R. R., Bagby, R. M., ... & Eaton, N. R. (2017). The Hierarchical Taxonomy of Psychopathology (HiTOP): A dimensional alternative to traditional nosologies. *Journal of abnormal psychology*, 126(4), 454.
- Kuhn, M. (2008). Building predictive models in R using the caret package. *Journal of statistical software*, 28(5), 1-26.
- Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*(Vol. 26). New York: Springer.
- Le Cessie, S., & Van Houwelingen, J. C. (1992). Ridge estimators in logistic regression. *Applied statistics*, 191-201.

- Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R news*, 2(3), 18-22.
- Mattera D. and Haykin S. 1999. Support vector machines for dynamic reconstruction of a chaotic system. In: Schölkopf B., Burges C.J.C., and Smola A.J. (Eds.), *Advances in Kernel Methods— Support Vector Learning*, MIT Press, Cambridge, MA, pp. 211– 242.
- Meehl, P. E. (1954). Clinical versus statistical prediction: A theoretical analysis and a review of the evidence.
- Meehl, P. E. (1956). Wanted—a good cook-book. *American Psychologist*, 11(6), 263.
- Meehl, P. E. (1957). When shall we use our heads instead of the formula?. *Journal of Counseling Psychology*, 4(4), 268.
- Meehl, P. E. (1986). Causes and effects of my disturbing little book. *Journal of personality assessment*, 50(3), 370-375.
- Meehl, P. E. (1992). Factors and taxa, traits and types, differences of degree and differences in kind. *Journal of personality*, 60(1), 117-174.
- Meehl, P. E., & Rosen, A. (1955). Antecedent probability and the efficiency of psychometric signs, patterns, or cutting scores. *Psychological bulletin*, 52(3), 194.
- Michalewicz, Z. (1994). GAs: What are they?. In *Genetic algorithms+ data structures= evolution programs* (pp. 13-30). Springer Berlin Heidelberg.
- Miller, A. (2002). *Subset selection in regression*. CRC Press.
- Mosier, C. I. (1951). The need and means of cross validation. I. Problems and designs of cross-validation. *Educational and Psychological Measurement*.
- Müller K.-R., Smola A., Rätsch G., Schölkopf B., Kohlmorgen J., and Vapnik V. 1997. Predicting time series with support vector machines. In: Gerstner W., Germond A.,

- Hasler M., and Nicoud J.-D. (Eds.), Artificial Neural Networks ICANN'97, Berlin. Springer Lecture Notes in Computer Science Vol. 1327 pp. 999–1004.
- R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.
- Ripley, B., Venables, B., Bates, D. M., Hornik, K., Gebhardt, A., & Firth, D. (2013). Package 'mass'. *CRAN R*.
- 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(1), 77.
- Sellers, K. F., & Shmueli, G. (2010). A flexible regression model for count data. *The Annals of Applied Statistics*, 4(2), 943-961.
- Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, 14(3), 199-222.
- Smola, A., & Vapnik, V. (1997). Support vector regression machines. *Advances in neural information processing systems*, 9, 155-161.
- Spiess, A. N., & Neumeyer, N. (2010). An evaluation of R^2 as an inadequate measure for nonlinear models in pharmacological and biochemical research: a Monte Carlo approach. *BMC pharmacology*, 10(1), 6.
- Statnikov, A., Wang, L., & Aliferis, C. F. (2008). A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. *BMC bioinformatics*, 9(1), 1.
- Steyerberg, E. (2009). *Clinical prediction models: a practical approach to development, validation, and updating*. Springer Science & Business Media.

- Stitson M., Gammerman A., Vapnik V., Vovk V., Watkins C., and Weston J. 1999. Support vector regression with ANOVA decomposition kernels. In: Schölkopf B., Burges C.J.C., and Smola A.J. (Eds.), *Advances in Kernel Methods—Support Vector Learning*, MIT Press Cambridge, MA, pp. 285–292.
- Tellegen, A., Ben-Porath, Y. S., McNulty, J. L., Arbisi, P. A., Graham, J. R., & Kaemmer, B. (2003). The MMPI-2 Restructured Clinical (RC) scales: Development, validation, and interpretation.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267-288.
- Thompson B. Stepwise regression and stepwise discriminant analysis need not apply here: a guidelines editorial. *Ed Psychol Meas* 1995;55:525–34.
- Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11), 1134-1142.
- Were, K., Bui, D. T., Dick, Ø. B., & Singh, B. R. (2015). A comparative assessment of support vector regression, artificial neural networks, and random forests for predicting and mapping soil organic carbon stocks across an Afromontane landscape. *Ecological Indicators*, 52, 394-403.
- Wickham, H. (2017). The tidyverse. *R package ver. 1.1. 1*.
- Wiggins, J. S. (1973). *Personality and prediction: Principles of personality assessment*. Menlo Park, CA: Addison Wesley.
- Wright, A. G., Pincus, A. L., & Lenzenweger, M. F. (2012). An empirical examination of distributional assumptions underlying the relationship between personality disorder

symptoms and personality traits. *Journal of Abnormal Psychology*, 121, 699 –706.

<http://dx.doi.org/10.1037/a0029042>

Yarkoni, T., & Westfall, J. (2017). Choosing prediction over explanation in psychology: Lessons from machine learning. *Perspectives on Psychological Science*, 12(6), 1100-1122.

Appendix: R Code

```
#####
# Code for Dissertation: Generalizability of Statistical Prediction #
# William Menton #
#####

require(tidyverse)
require(haven)
require(gridExtra)
require(grid)
require(caret)
require(MASS)
require(glmnet)
require(randomForest)
require(pROC)
require(corr)

# Initialization
#-----

# Set working directory -----
wd <- # Add working directory here
setwd(wd)
outwd <- paste(wd, "/Output", sep = "")

# Functions
#-----

# Function to add MMPI-2-RF validity status indicators to dataset.
validate <- function(data){
  new_data <- mutate(data, consistent_rf = 1)
  new_data <- mutate(new_data, valid_rf = 1)

  for(i in 1:nrow(new_data)){
    if(new_data[i, "rfCNS"] >= 18 ||
       new_data[i, "LTrfVRIN"] >= 80 ||
       new_data[i, "LTrfTRIN"] >= 80){
      new_data[i, "consistent_rf"] <- 0
    }
    if(new_data[i, "consistent_rf"] == 0 ||
       new_data[i, "LTrfFp"] >= 100 ||
       new_data[i, "LTrfF"] >= 120){
      new_data[i, "valid_rf"] <- 0
    }
  }
  return(new_data)
}
```

```

# Function to extract cases with valid MMPI-2-RF protocols. Calls validate().
valid_only <- function(data){
  new_data <- validate(data)
  new_data <- filter(new_data, valid_rf == 1)
  return(new_data)
}

# Function to calculate base rates. Assumes group of interest is
# coded 1, else 0. The name of the grouping variable should be supplied
# as a string.
calc_br <- function(data, varname){
  obs <- data[, varname]
  base_freq <- which(obs == 1) %>%
    length()
  comp_freq <- which(obs == 0) %>%
    length()
  base_rate <- base_freq / (base_freq + comp_freq)
  return(base_rate)
}

# Function to randomize data and create K-fold cross-validation indices. Used as a
# helper
# for other functions.
cv.ind <- function(data, K=K){
  rand.data <- data[sample(nrow(data)),] # Shuffle dataset
  fold.size <- round(nrow(rand.data) / K) # Set fold size
  cv.indices <- matrix(1, nrow=nrow(rand.data), ncol=K) # matrix to hold index values

  for(k in 1:K){ # Index test data as 0
    fold.start <- (fold.size * (k - 1)) + 1

    if(k == K){
      fold.end <- nrow(rand.data)
    } else {
      fold.end <- fold.start + fold.size - 1
    }

    cv.indices[fold.start:fold.end, k] <- 0
  }

  cv.out <- list(rand.data = rand.data, cv.indices = cv.indices)
  return(cv.out)
}

# Function to take a cv.glmnet object and return cross-validation RMSE associated with
# the best cross-validated lambda (associated with lowest MSE)
glmnet_cv_rmse <- function mdl){
  best_lambda <- mdl$lambda.min
  best_idx <- which(mdl$lambda == best_lambda)
  RMSE <- sqrt(mdl$cvm[best_idx])
  return(RMSE)
}

# Function to take a cv.glmnet object and return cross-validation R2 associated with
# the best cross-validated lambda. Note that this involves an additional cross-
# validation
# process (these are not the folds originally used to train the full glmnet model).
# May not work if alpha was not specified in original glmnet call. Calls cv.ind()
# and R2()
glmnet_cv_R2 <- function(mdl, x, y, K = 10){

```

```

# mdl = glmnet model
# x = training data for predictor variables
# y = training data for criterion variable
# K = number of folds for k-fold cross-validation

best_lambda <- mdl$lambda.min
alph <- mdl$glmnet.fit$call$alpha

data <- cbind(y, x)
folds <- cv.ind(data, K = K) # Get folds
cv_data <- folds$rand.data # Shuffled data
cv_idx <- folds$cv.indices # Matrix containing indices for each fold

# Separate data back into predictor and criterion objects for further glmnet calls
full_preds <- cv_data[, -1]
full_crit <- cv_data[, 1]

# Create vector to hold R2 values
R2s <- rep(NA, K)

# Loop over K folds, training new glmnet model and determining R2 against the held out
data
for (j in 1:K){
  curr_idx <- cv_idx[, j]
  train_preds <- full_preds[curr_idx == 1, ]
  train_crit <- full_crit[curr_idx == 1]
  test_preds <- full_preds[curr_idx == 0, ]
  test_crit <- full_crit[curr_idx == 0]

  cv_mdl <- glmnet(x = train_preds, y = train_crit, alpha = alph, lambda =
best_lambda)
  test_vals <- predict(cv_mdl, s = cv_mdl$lambda.min, newx = test_preds)
  test_R2 <- R2(test_vals, test_crit)
  R2s[j] <- test_R2
}
return(mean(R2s))
}

# Function to take a cv.glmnet object and return cross-validated accuracy and AUC
associated with
# the best cross-validated lambda. Note that this involves an additional cross-
validation
# process (these are not the folds originally used to train the full glmnet model).
# May not work if alpha was not specified in original glmnet call. Calls cv.ind(),
roc(), and
# confusionMatrix(). Only works for logistic models
glmnet_cv_class_stats <- function(mdl, x, y, K = 10){
  # mdl = glmnet model with family = "binomial"
  # x = training data for predictor variables
  # y = training data for criterion variable
  # K = number of folds for k-fold cross-validation

  best_lambda <- mdl$lambda.min
  alph <- mdl$glmnet.fit$call$alpha

  data <- cbind(y, x) %>% as.data.frame()
  folds <- cv.ind(data, K = K) # Get folds
  cv_data <- folds$rand.data # Shuffled data
  cv_idx <- folds$cv.indices # Matrix containing indices for each fold

  # Separate shuffled data back into predictor and criterion objects for further
glmnet calls

```

```

full_preds <- cv_data[, -1] %>% as.matrix
full_crit <- cv_data[, 1]

# Create vectors to hold accuracy and AUC values
Accs <- rep(NA, K)
AUCs <- rep(NA, K)

# Loop over K folds, training new glmnet models and evaluating performance against
the held out data
for (j in 1:K){
  curr_idx <- cv_idx[, j]
  test_crit <- full_crit[curr_idx == 0]
  num_crit <- as.numeric(test_crit) # convert factor to numeric to evaluate number
of positive cases
  n_pos <- which(num_crit == 2) %>% length()

  if(n_pos == 0){ # Return NAs if no positive cases exist in test data for this fold
    Accs[j] <- NA
    AUCs[j] <- NA
  } else{
    train_preds <- full_preds[curr_idx == 1, ]
    train_crit <- full_crit[curr_idx == 1]
    test_preds <- full_preds[curr_idx == 0, ]

    cv_mdl <- glmnet(x = as.matrix(train_preds), y = train_crit, alpha = alph,
lambda = best_lambda,
                      family = "binomial")
    test_pred_probs <- predict(cv_mdl, s = cv_mdl$lambda.min, newx = test_preds,
                              type = "response")
    test_pred_class <- predict(cv_mdl, s = cv_mdl$lambda.min, newx = test_preds,
                              type = "class")
    test_acc <- confusionMatrix(test_pred_class, test_crit)$overall['Accuracy']
    test_auc <- roc(test_crit, test_pred_probs)$auc

    Accs[j] <- test_acc
    AUCs[j] <- test_auc
  }
}
return(list(Acc = mean(Accs, na.rm = TRUE),
           AUC = mean(AUCs, na.rm = TRUE)))
}

# Extracts the name of the response (outcome) variable from a glm formula.
# From StackOverflow.
getResponse <- function(formula) {
  tt <- terms(formula)
  vars <- as.character(attr(tt, "variables"))[-1] ## [1] is the list call
  response <- attr(tt, "response") # index of response var
  vars[response]
}

# Function to return cross-validated R2 and RMSE for a forwards-backwards stepAIC
object.
# Calls cv.ind().
step_cv_reg <- function(lo_mdl, hi_mdl, data, K = 10){
  # Arguments:
  # lo_mdl = a glm object defining the lower model for the stepwise search
  # hi_mdl = a glm object defining the upper model for the stepwise search
  # data = data
  # K = number of folds for k-fold cross-validation

  lo_form <- eval(lo_mdl$call[[2]])

```

```

hi_form <- eval(hi_mdl$call[[2]])
crit_nam <- names(hi_mdl$model)[1]

folds <- cv.ind(data, K = K)
cv_data <- folds$rand.data # Shuffled data
cv_idx <- folds$cv.indices # Matrix containing indices for each fold

# Create vectors to hold accuracy and AUC values
R2s <- rep(NA, K)
RMSEs <- rep(NA, K)

for(i in 1:K){
  curr_idx <- cv_idx[, i]
  train <- data[curr_idx == 1, ]
  test <- data[curr_idx == 0, ]

  fit_lo <- lm(lo_form, data = train)
  fit_hi <- lm(hi_form, data = train)
  fit_step <- stepAIC(object = fit_lo,
                     scope = list(upper = fit_hi, lower = fit_lo),
                     trace = FALSE)

  obs_out <- test[, crit_nam]
  pred_out <- predict(fit_step, test)
  R2s[i] <- R2(pred_out, obs_out)
  RMSEs[i] <- RMSE(pred_out, obs_out)
}
return(list(R2 = mean(R2s, na.rm = TRUE),
           RMSE = mean(RMSEs, na.rm = TRUE)))
}

# Function to return cross-validated accuracy and AUC for a forwards-backwards stepAIC
# logistic regression object. Assumes outcome variable is a two-level factor. Calls
# cv.ind().
step_cv_class <- function(lo_mdl, hi_mdl, data, K = 10){
  # Arguments:
  #   lo_mdl = a glm object defining the lower model for the stepwise search
  #   hi_mdl = a glm object defining the upper model for the stepwise search
  #   data = data
  #   K = number of folds for k-fold cross-validation

  lo_form <- eval(lo_mdl$call[[2]])
  hi_form <- eval(hi_mdl$call[[2]])
  crit_nam <- names(hi_mdl$model)[1]

  folds <- cv.ind(data, K = K)
  cv_data <- folds$rand.data # Shuffled data
  cv_idx <- folds$cv.indices # Matrix containing indices for each fold

  # Create vectors to hold accuracy and AUC values
  Accs <- rep(NA, K)
  AUCs <- rep(NA, K)

  for(i in 1:K){
    curr_idx <- cv_idx[, i]
    train <- data[curr_idx == 1, ]
    test <- data[curr_idx == 0, ]

    fit_lo <- glm(lo_form, data = train, family = "binomial")
    fit_hi <- glm(hi_form, data = train, family = "binomial")
    fit_step <- stepAIC(object = fit_lo,

```



```

        scope = list(upper = fit_hi, lower = fit_lo),
        trace = FALSE)

obs_out <- test[, crit_nam] %>% unlist() %>% as.factor()
n_pos <- which(as.numeric(obs_out) == 2) %>% length()

if(n_pos == 0){ # Return NAs if no positive cases exist in test data for this fold
  Accs[i] <- NA
  AUCs[i] <- NA
} else{
  pred_probs <- predict(fit_step, test, type = "response")
  pred_class <- rep(0, nrow(test))
  pred_class[pred_probs >= .5] = 1
  pred_class <- as.factor(pred_class)
  levels(pred_class) <- levels(obs_out)

  test_acc <- confusionMatrix(pred_class, obs_out)$overall['Accuracy']
  test_auc <- roc(obs_out, pred_probs)$auc

  Accs[i] <- test_acc
  AUCs[i] <- test_auc
}
}
return(list(Acc = mean(Accs, na.rm = TRUE),
           AUC = mean(AUCs, na.rm = TRUE)))
}

# Function to return AUC values for all RC Scales calculated
# against a single dichotomous grouping variable.
# Arguments:
#   data = dataset
#   group = dichotomous grouping variable (string)
rc_auc <- function(data, group){
  rf_scales <- c("rfRCd", "rfRC1", "rfRC2",
                "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8", "rfRC9")

  AUCs <- matrix(NA, nrow = length(rf_scales), ncol = 2) %>%
    data.frame
  colnames(AUCs) <- c("Scale", "AUC")
  AUCs[, 1] <- rf_scales

  for(i in 1:length(rf_scales)){
    curr_scale <- rf_scales[i]
    resp <- data[[group]]
    pred <- data[[curr_scale]]

    roc_obj <- roc(resp, pred)

    AUCs[i, 2] <- roc_obj$auc
  }
  return(AUCs)
}

# Function to return AUC values for all MMPI-2-RF scales calculated
# against a single dichotomous grouping variable.
# Arguments:
#   data = dataset
#   group = dichotomous grouping variable (string)
rf_auc <- function(data, group){
  rf_scales <- c("rfVRIN", "rfTRIN", "rfF", "rfFp", "rfFs", "rfFBS", "rfRBS", "rfL",
                "rfK",
                "rfEID", "rfTHD", "rfBXD", "rfRCd", "rfRC1", "rfRC2",
                "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8", "rfRC9",

```

```

      "rfMLS", "rfGIC", "rfHPC", "rfNUC", "rfCOG",
      "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY",
      "rfANP", "rfBRF", "rfMSF",
      "rfJCP", "rfSUB", "rfAGG", "rfACT",
      "rfFML", "rfIPP", "rfSAV", "rfSHY", "rfDSF",
      "rfAES", "rfMEC",
      "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")

AUCs <- matrix(NA, nrow = length(rf_scales), ncol = 2) %>%
  data.frame
colnames(AUCs) <- c("Scale", "AUC")
AUCs[, 1] <- rf_scales

for(i in 1:length(rf_scales)){
  curr_scale <- rf_scales[i]
  resp <- data[[group]]
  pred <- data[[curr_scale]]

  roc_obj <- roc(resp, pred)

  AUCs[i, 2] <- roc_obj$auc
}
return(AUCs)
}

# Function to generate plot of regression model fit
# Arguments:
#   obs = vector of observed values
#   pred = vector of predicted values
#   title = plot title (optional)
fit_plot <- function(obs, pred, title = "Observed vs. Predicted Values"){
  dat <- cbind(obs, pred) %>% as.data.frame
  plotted <- ggplot(dat, aes(x = dat[,1], y = dat[,2])) +
    geom_point(alpha = .1) +
    geom_jitter(width = .05, height = 0) +
    geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
    labs(x = "Observed", y = "Predicted") +
    ggtitle(title) +
    theme(plot.title = element_text(hjust = .5))
  return(plotted)
}

# Function to generate residual plots
# Arguments:
#   obs = vector of observed values
#   pred = vector of predicted values
#   title = plot title (optional)
resid_plot <- function(obs, pred, title = "Residual Plot"){
  dat <- cbind(obs, (obs - pred)) %>% as.data.frame
  plotted <- ggplot(dat, aes(x = dat[,1], y = dat[,2])) +
    geom_point(alpha = .1) +
    geom_jitter(width = .05, height = 0) +
    geom_hline(yintercept = 0, linetype = "dashed") +
    labs(x = "Observed", y = "Residual") +
    ggtitle(title) +
    theme(plot.title = element_text(hjust = .5))
  return(plotted)
}

# Clean and prepare data
#-----

```

```

# Read in data -----
MN_inpt_full <- read_spss("Inpatient - HCMC and VA - dissertation2.sav")
ppath_full <- read_spss("Portage Path - dissertation.sav")

# Create diagnostic groups -----
dep_dx <- c(29620:29636, 30040, 31100) # MDD, Dysthymia, and Dep NOS, respectively
sub_dx <- 30300:30590 # Substance dependence or abuse
thought_dx <- c(29510:29590, 29890) # Schizophrenia, schizoaffective,
schizophreniform, psych d/o NOS
som_dx <- c(30070:30082, 30011) # hypochondriasis, somatoform, somatization,
conversion

# Add diagnostic group codings to data-----

# Add codes to inpatient data
MN_inpt_full <- mutate(MN_inpt_full, dep_do = 0) %>%
  mutate(sub_do = 0) %>%
  mutate(thought_do = 0) %>%
  mutate(som_do = 0)

MN_inpt_full <- MN_inpt_full %>%
  replace_na(list(dai1q = 9, dai2q = 9, dai3q = 9, dai4q = 9))

for(i in 1:nrow(MN_inpt_full)){
  if((MN_inpt_full[i, "dai1"] %in% dep_dx) && MN_inpt_full[i, "dai1q"] != 1){
    MN_inpt_full[i, "dep_do"] <- 1
  }
  if((MN_inpt_full[i, "dai2"] %in% dep_dx) && MN_inpt_full[i, "dai2q"] != 1){
    MN_inpt_full[i, "dep_do"] <- 1
  }
  if((MN_inpt_full[i, "dai3"] %in% dep_dx) && MN_inpt_full[i, "dai3q"] != 1){
    MN_inpt_full[i, "dep_do"] <- 1
  }
  if((MN_inpt_full[i, "dai4"] %in% dep_dx) && MN_inpt_full[i, "dai4q"] != 1){
    MN_inpt_full[i, "dep_do"] <- 1
  }
  if((MN_inpt_full[i, "dai1"] %in% sub_dx) && MN_inpt_full[i, "dai1q"] != 1){
    MN_inpt_full[i, "sub_do"] <- 1
  }
  if((MN_inpt_full[i, "dai2"] %in% sub_dx) && MN_inpt_full[i, "dai2q"] != 1){
    MN_inpt_full[i, "sub_do"] <- 1
  }
  if((MN_inpt_full[i, "dai3"] %in% sub_dx) && MN_inpt_full[i, "dai3q"] != 1){
    MN_inpt_full[i, "sub_do"] <- 1
  }
  if((MN_inpt_full[i, "dai4"] %in% sub_dx) && MN_inpt_full[i, "dai4q"] != 1){
    MN_inpt_full[i, "sub_do"] <- 1
  }
  if((MN_inpt_full[i, "dai1"] %in% thought_dx) && MN_inpt_full[i, "dai1q"] != 1){
    MN_inpt_full[i, "thought_do"] <- 1
  }
  if((MN_inpt_full[i, "dai2"] %in% thought_dx) && MN_inpt_full[i, "dai2q"] != 1){
    MN_inpt_full[i, "thought_do"] <- 1
  }
  if((MN_inpt_full[i, "dai3"] %in% thought_dx) && MN_inpt_full[i, "dai3q"] != 1){
    MN_inpt_full[i, "thought_do"] <- 1
  }
  if((MN_inpt_full[i, "dai4"] %in% thought_dx) && MN_inpt_full[i, "dai4q"] != 1){
    MN_inpt_full[i, "thought_do"] <- 1
  }
  if((MN_inpt_full[i, "dai1"] %in% som_dx) && MN_inpt_full[i, "dai1q"] != 1){
    MN_inpt_full[i, "som_do"] <- 1
  }
}

```

```

  if((MN_inpt_full[i, "dai2"] %in% som_dx) && MN_inpt_full[i, "dai2q"] != 1){
    MN_inpt_full[i, "som_do"] <- 1
  }
  if((MN_inpt_full[i, "dai3"] %in% som_dx) && MN_inpt_full[i, "dai3q"] != 1){
    MN_inpt_full[i, "som_do"] <- 1
  }
  if((MN_inpt_full[i, "dai4"] %in% som_dx) && MN_inpt_full[i, "dai4q"] != 1){
    MN_inpt_full[i, "som_do"] <- 1
  }
}

# Add codes to Portage Path
ppath_full <- mutate(ppath_full, dep_do = 0) %>%
  mutate(sub_do = 0) %>%
  mutate(thought_do = 0) %>%
  mutate(som_do = 0)

for(i in 1:nrow(ppath_full)){
  if((ppath_full[i, "ai1"] %in% dep_dx) && ppath_full[i, "ai1ro"] != 1){
    ppath_full[i, "dep_do"] <- 1
  }
  if((ppath_full[i, "ai2"] %in% dep_dx) && ppath_full[i, "ai2ro"] != 1){
    ppath_full[i, "dep_do"] <- 1
  }
  if((ppath_full[i, "ai3"] %in% dep_dx) && ppath_full[i, "ai3ro"] != 1){
    ppath_full[i, "dep_do"] <- 1
  }
  if((ppath_full[i, "ai1"] %in% sub_dx) && ppath_full[i, "ai1ro"] != 1){
    ppath_full[i, "sub_do"] <- 1
  }
  if((ppath_full[i, "ai2"] %in% sub_dx) && ppath_full[i, "ai2ro"] != 1){
    ppath_full[i, "sub_do"] <- 1
  }
  if((ppath_full[i, "ai3"] %in% sub_dx) && ppath_full[i, "ai3ro"] != 1){
    ppath_full[i, "sub_do"] <- 1
  }
  if((ppath_full[i, "ai1"] %in% thought_dx) && ppath_full[i, "ai1ro"] != 1){
    ppath_full[i, "thought_do"] <- 1
  }
  if((ppath_full[i, "ai2"] %in% thought_dx) && ppath_full[i, "ai2ro"] != 1){
    ppath_full[i, "thought_do"] <- 1
  }
  if((ppath_full[i, "ai3"] %in% thought_dx) && ppath_full[i, "ai3ro"] != 1){
    ppath_full[i, "thought_do"] <- 1
  }
  if((ppath_full[i, "ai1"] %in% som_dx) && ppath_full[i, "ai1ro"] != 1){
    ppath_full[i, "som_do"] <- 1
  }
  if((ppath_full[i, "ai2"] %in% som_dx) && ppath_full[i, "ai2ro"] != 1){
    ppath_full[i, "som_do"] <- 1
  }
  if((ppath_full[i, "ai3"] %in% som_dx) && ppath_full[i, "ai3ro"] != 1){
    ppath_full[i, "som_do"] <- 1
  }
}

# Create numeric copies of group variables prior to conversion to factors
MN_inpt_full <- mutate(MN_inpt_full, dep_do_num = dep_do) %>%
  mutate(sub_do_num = sub_do) %>%
  mutate(thought_do_num = thought_do) %>%
  mutate(som_do_num = som_do)
ppath_full <- mutate(ppath_full, dep_do_num = dep_do) %>%
  mutate(sub_do_num = sub_do) %>%

```

```

mutate(thought_do_num = thought_do) %>%
mutate(som_do_num = som_do)

# Coerce disorder group variables to factors
MN_inpt_full$dep_do <- as.factor(MN_inpt_full$dep_do)
MN_inpt_full$sub_do <- as.factor(MN_inpt_full$sub_do)
MN_inpt_full$thought_do <- as.factor(MN_inpt_full$thought_do)
MN_inpt_full$som_do <- as.factor(MN_inpt_full$som_do)
ppath_full$dep_do <- as.factor(ppath_full$dep_do)
ppath_full$sub_do <- as.factor(ppath_full$sub_do)
ppath_full$thought_do <- as.factor(ppath_full$thought_do)
ppath_full$som_do <- as.factor(ppath_full$som_do)

# Rename disorder group variable levels so they will play nicely with caret
levels(MN_inpt_full$dep_do) <- c("Neg", "Pos")
levels(MN_inpt_full$sub_do) <- c("Neg", "Pos")
levels(MN_inpt_full$thought_do) <- c("Neg", "Pos")
levels(MN_inpt_full$som_do) <- c("Neg", "Pos")
levels(ppath_full$dep_do) <- c("Neg", "Pos")
levels(ppath_full$sub_do) <- c("Neg", "Pos")
levels(ppath_full$thought_do) <- c("Neg", "Pos")
levels(ppath_full$som_do) <- c("Neg", "Pos")

# Create final datasets using only cases with valid MMPI-2-RF protocols
HCMC <- valid_only(MN_inpt_full) %>%
  filter(site == 1)
VAMC <- valid_only(MN_inpt_full) %>%
  filter(site == 2)
ppath <- valid_only(ppath_full)

#=====
# REGRESSION ANALYSES
#=====

# Set training control parameters
cv_control <- trainControl(method = "cv", number = 10)

# Set model formulas for repeated use
dep_form <- as.formula("msefct1 ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")
sub_form <- as.formula("msefct4 ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")
tht_form <- as.formula("msefct5 ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")
pai_form <- as.formula("msefct8 ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")

# OLS Linear Regression
#=====

# Train models using full training data
set.seed(1001)
dep_lm_train <- train(form = dep_form, data = HCMC,
                      trControl = cv_control, method = "lm")
set.seed(1002)
sub_lm_train <- train(form = sub_form, data = HCMC,
                      trControl = cv_control, method = "lm")
set.seed(1003)
tht_lm_train <- train(form = tht_form, data = HCMC,
                      trControl = cv_control, method = "lm")
set.seed(1004)

```

```

pai_lm_train <- train(form = pai_form, data = HCMC,
                      trControl = cv_control, method = "lm")

# Get training data RMSEs for full models
dep_lm_train_rmse <- RMSE(predict(dep_lm_train, HCMC), HCMC$msefct1)
sub_lm_train_rmse <- RMSE(predict(sub_lm_train, HCMC), HCMC$msefct4)
tht_lm_train_rmse <- RMSE(predict(tht_lm_train, HCMC), HCMC$msefct5)
pai_lm_train_rmse <- RMSE(predict(pai_lm_train, HCMC), HCMC$msefct8)

# Export OLS Linear Regression results
lm_mdls <- list(dep_lm_train, sub_lm_train, tht_lm_train, pai_lm_train)

lm_out <- matrix(NA,
                nrow = length(lm_mdls[[1]]$finalModel$coefficients) + 7,
                ncol = length(lm_mdls))
rownames(lm_out) <- c(names(lm_mdls[[1]]$finalModel$coefficients), "R2-train",
                    "Adj R2 train", "RMSE-train", "R2-CV", "RMSE-CV",
                    "R2-EV", "RMSE-EV")
colnames(lm_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(lm_mdls)){
  curr_mdl <- lm_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- summary(curr_mdl)$r.squared
  adj_R2_train <- summary(curr_mdl)$adj.r.squared
  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  R2_cv <- curr_mdl$results$Rsquared
  rmse_cv <- curr_mdl$results$RMSE
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  lm_out[1:n_coefs, i] <- coefs
  lm_out[n_coefs + 1, i] <- R2_train
  lm_out[n_coefs + 2, i] <- adj_R2_train
  lm_out[n_coefs + 3, i] <- rmse_train
  lm_out[n_coefs + 4, i] <- R2_cv
  lm_out[n_coefs + 5, i] <- rmse_cv
  lm_out[n_coefs + 6, i] <- R2_ev
  lm_out[n_coefs + 7, i] <- rmse_ev
}

# Export OLS linear regression results
setwd(outwd)
write.csv(lm_out, "OLS Linear Regression Output.csv")
setwd(wd)

# Negative Binomial Regression
#####

# Set link function to "log" (recommended for modeling count data)
tune_glm.nb <- as.data.frame("log")
names(tune_glm.nb) <- "link"

```

```

# Train models using full training data
set.seed(2001)
dep_nb_train <- train(form = dep_form, data = HCMC,
                      trControl = cv_control, method = "glm.nb",
                      tuneGrid = tune_glm.nb)

set.seed(2002)
sub_nb_train <- train(form = sub_form, data = HCMC,
                      trControl = cv_control, method = "glm.nb",
                      tuneGrid = tune_glm.nb)

set.seed(2003)
tht_nb_train <- train(form = tht_form, data = HCMC,
                      trControl = cv_control, method = "glm.nb",
                      tuneGrid = tune_glm.nb)

set.seed(2004)
pai_nb_train <- train(form = pai_form, data = HCMC,
                      trControl = cv_control, method = "glm.nb",
                      tuneGrid = tune_glm.nb)

# Export negative binomial regression results
nb_mdls <- list(dep_nb_train, sub_nb_train, tht_nb_train, pai_nb_train)

nb_out <- matrix(NA,
                nrow = length(nb_mdls[[1]]$finalModel$coefficients) + 3,
                ncol = length(nb_mdls))
rownames(nb_out) <- c(names(nb_mdls[[1]]$finalModel$coefficients),
                     "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(nb_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(nb_mdls)){
  curr_mdl <- nb_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  rmse_cv <- curr_mdl$results$RMSE
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  nb_out[1:n_coefs, i] <- coefs
  nb_out[n_coefs + 1, i] <- rmse_train
  nb_out[n_coefs + 2, i] <- rmse_cv
  nb_out[n_coefs + 3, i] <- rmse_ev
}

# Export negative binomial regression results
setwd(outwd)
write.csv(nb_out, "Negative Binomial Regression Output.csv")
setwd(wd)

# Ridge Regression - glmnet
#=====

# Train models using full training data
preds <- c("rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8",
"rfRC9")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

```

```

set.seed(3001)
dep_ridge_train <- cv.glmnet(x = train_mat, y = HCMC$msefct1, alpha = 0)
set.seed(3002)
sub_ridge_train <- cv.glmnet(x = train_mat, y = HCMC$msefct4, alpha = 0)
set.seed(3003)
tht_ridge_train <- cv.glmnet(x = train_mat, y = HCMC$msefct5, alpha = 0)
set.seed(3004)
pai_ridge_train <- cv.glmnet(x = train_mat, y = HCMC$msefct8, alpha = 0)

# Export ridge regression results
ridge_mdls <- list(dep_ridge_train, sub_ridge_train, tht_ridge_train, pai_ridge_train)

ridge_out <- matrix(NA,
  nrow = nrow(ridge_mdls[[1]]$glmnet.fit$beta) + 7,
  ncol = length(ridge_mdls))
rownames(ridge_out) <- c("Intercept", rownames(ridge_mdls[[1]]$glmnet.fit$beta),
  "R2-train", "R2-CV", "R2-EV",
  "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(ridge_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

test_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()

for(i in 1:length(ridge_mdls)){
  curr_mdl <- ridge_mdls[[i]]
  lambda_idx <- which(curr_mdl$glmnet.fit$lambda == curr_mdl$lambda.min)
  coefs <- curr_mdl$glmnet.fit$beta[, lambda_idx]
  n_coefs <- length(coefs)

  train_outcome <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat)
  test_pred <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_mat)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- R2(train_outcome, HCMC[, reg_outcomes[i]])
  R2_cv <- glmnet_cv_R2(curr_mdl, x = train_mat, y = as.matrix(HCMC[,
reg_outcomes[i]]))
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_train <- RMSE(train_outcome, HCMC[, reg_outcomes[i]])
  rmse_cv <- glmnet_cv_rmse(curr_mdl)
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  ridge_out[1, i] <- curr_mdl$glmnet.fit$a0[lambda_idx]
  ridge_out[2:(n_coefs + 1), i] <- coefs
  ridge_out[n_coefs + 2, i] <- R2_train
  ridge_out[n_coefs + 3, i] <- R2_cv
  ridge_out[n_coefs + 4, i] <- R2_ev
  ridge_out[n_coefs + 5, i] <- rmse_train
  ridge_out[n_coefs + 6, i] <- rmse_cv
  ridge_out[n_coefs + 7, i] <- rmse_ev
}

setwd(outwd)
write.csv(ridge_out, "Ridge Regression Output.csv")
setwd(wd)

# Random Forests Regression
#=====

# Train models using full training data

```



```

preds <- c("rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8",
"rfRC9")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

# Define random forests training control. This will allow the algorithm to explore a
# greater range of tuning parameters.
rf_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random")

set.seed(4001)
dep_rf_train <- train(form = dep_form, data = HCMC,
                      trControl = rf_control, method = "rf",
                      tuneLength = 15)

set.seed(4002)
sub_rf_train <- train(form = sub_form, data = HCMC,
                      trControl = rf_control, method = "rf",
                      tuneLength = 15)

set.seed(4003)
tht_rf_train <- train(form = tht_form, data = HCMC,
                      trControl = rf_control, method = "rf",
                      tuneLength = 15)

set.seed(4004)
pai_rf_train <- train(form = pai_form, data = HCMC,
                      trControl = rf_control, method = "rf",
                      tuneLength = 15)

# Export random forests regression results
rf_mdls <- list(dep_rf_train, sub_rf_train, tht_rf_train, pai_rf_train)

rf_out <- matrix(NA,
                nrow = length(rf_mdls[[1]]$finalModel$importance) + 3,
                ncol = length(rf_mdls))
rownames(rf_out) <- c(rownames(rf_mdls[[1]]$finalModel$importance),
                      "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(rf_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(rf_mdls)){
  curr_mdl <- rf_mdls[[i]]
  importance <- curr_mdl$finalModel$importance
  n_coefs <- length(importance)

  best_mtry <- curr_mdl$bestTune %>% as.numeric()
  best_idx <- which(curr_mdl$results$mtry == best_mtry)
  best_mtry_RMSE <- curr_mdl$results$RMSE[best_idx]

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  rmse_cv <- best_mtry_RMSE
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  rf_out[1:n_coefs, i] <- importance
  rf_out[n_coefs + 1, i] <- rmse_train
  rf_out[n_coefs + 2, i] <- rmse_cv
  rf_out[n_coefs + 3, i] <- rmse_ev
}

# Export random forests regression results

```

```

setwd(outwd)
write.csv(rf_out, "Random Forests Regression Output.csv")
setwd(wd)

# Support vector regression
#=====

# Train models using full set of training data
set.seed(6001)
dep_svr_train <- train(form = dep_form, data = HCMC,
                       preProc = c("center", "scale"),
                       trControl = cv_control, method = "svmRadial",
                       tuneLength = 14)

set.seed(6002)
sub_svr_train <- train(form = sub_form, data = HCMC,
                       preProc = c("center", "scale"),
                       trControl = cv_control, method = "svmRadial",
                       tuneLength = 14)

set.seed(6003)
tht_svr_train <- train(form = tht_form, data = HCMC,
                       preProc = c("center", "scale"),
                       trControl = cv_control, method = "svmRadial",
                       tuneLength = 14)

set.seed(6004)
pai_svr_train <- train(form = pai_form, data = HCMC,
                       preProc = c("center", "scale"),
                       trControl = cv_control, method = "svmRadial",
                       tuneLength = 14)

# Evaluate SVR models on external data and consolidate results

svr_mdls <- list(dep_svr_train, sub_svr_train, tht_svr_train, pai_svr_train)

svr_out <- matrix(NA,
                 nrow = 3,
                 ncol = length(svr_mdls))
rownames(svr_out) <- c("RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(svr_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(svr_mdls)){
  curr_mdl <- svr_mdls[[i]]

  best_C <- curr_mdl$bestTune$C
  best_idx <- which(curr_mdl$results$C == best_C)
  best_C_RMSE <- curr_mdl$results$RMSE[best_idx]

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  rmse_train <- RMSE(predict(curr_mdl, HCMC), train_outcome)
  rmse_cv <- best_C_RMSE
  rmse_ev <- RMSE(test_pred, test_outcome)

  svr_out[1, i] <- rmse_train
  svr_out[2, i] <- rmse_cv
  svr_out[3, i] <- rmse_ev
}

# Export SVR results

```

```

setwd(outwd)
write.csv(svr_out, "Support Vector Regression Output.csv")
setwd(wd)

#=====
# CLASSIFICATION ANALYSES
#=====

# Set standard training control parameters
acc_control <- trainControl(method = "cv", number = 10, savePredictions = TRUE,
                           classProbs = TRUE)
roc_control <- trainControl(method = "cv", number = 10, savePredictions = TRUE,
                           classProbs = TRUE, summaryFunction = twoClassSummary)

# Set model formulas for repeated use
dep_class_form <- as.formula("dep_do ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")
sub_class_form <- as.formula("sub_do ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")
tht_class_form <- as.formula("thought_do ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 +
rfRC6 + rfRC7 + rfRC8 + rfRC9")
som_class_form <- as.formula("som_do ~ rfRCd + rfRC1 + rfRC2 + rfRC3 + rfRC4 + rfRC6 +
rfRC7 + rfRC8 + rfRC9")

# Multivariate logistic regression
#=====

# Train models using full training data
set.seed(7001)
dep_lr_train <- train(form = dep_class_form, data = HCMC,
                    trControl = acc_control, method = "glm")
set.seed(7002)
sub_lr_train <- train(form = sub_class_form, data = HCMC,
                    trControl = acc_control, method = "glm")
set.seed(7003)
tht_lr_train <- train(form = tht_class_form, data = HCMC,
                    trControl = acc_control, method = "glm")
set.seed(7004)
som_lr_train <- train(form = som_class_form, data = HCMC,
                    trControl = acc_control, method = "glm")

# Recompute models using to obtain cross-validate ROC AUC (caret does not
# simultaneously calculate Accuracy and ROC)
set.seed(7001)
dep_lr_roc <- train(form = dep_class_form, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")
set.seed(7002)
sub_lr_roc <- train(form = sub_class_form, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")
set.seed(7003)
tht_lr_roc <- train(form = tht_class_form, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")
set.seed(7004)
som_lr_roc <- train(form = som_class_form, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")

# Test models on external data and consolidate results
lr_acc_mdls <- list(dep_lr_train, sub_lr_train, tht_lr_train, som_lr_train)

```

```

lr_roc_mdls <- list(dep_lr_roc, sub_lr_roc, tht_lr_roc, som_lr_roc)

lr_out <- matrix(NA,
                nrow = length(lr_acc_mdls[[1]]$finalModel$coefficients) + 8,
                ncol = length(lr_acc_mdls))
rownames(lr_out) <- c(names(lr_acc_mdls[[1]]$finalModel$coefficients),
                    "Acc-Train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                    "AUC-Train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(lr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(lr_acc_mdls)){
  curr_mdl <- lr_acc_mdls[[i]]
  curr_roc <- lr_roc_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_cv <- curr_mdl$results$Accuracy
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_roc$results$ROC
  AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
  AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

  lr_out[1:n_coefs, i] <- coefs
  lr_out[n_coefs + 1, i] <- acc_train
  lr_out[n_coefs + 2, i] <- acc_cv
  lr_out[n_coefs + 3, i] <- acc_ev_VAMC
  lr_out[n_coefs + 4, i] <- acc_ev_PP
  lr_out[n_coefs + 5, i] <- AUC_train
  lr_out[n_coefs + 6, i] <- AUC_cv
  lr_out[n_coefs + 7, i] <- AUC_ev_VAMC
  lr_out[n_coefs + 8, i] <- AUC_ev_PP
}

# Export logistic regression results
setwd(outwd)
write.csv(lr_out, "Multivariate Logistic Regression Output.csv")
setwd(wd)

# Ridge Logistic Regression - glmnet
=====

# Train models using full training data
preds <- c("rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8",
"rfRC9")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%

```

```

  as.matrix()
set.seed(9001)
dep_rlr_train <- cv.glmnet(x = train_mat, y = HCMC$dep_do, alpha = 0,
                           family = "binomial")

set.seed(9002)
sub_rlr_train <- cv.glmnet(x = train_mat, y = HCMC$sub_do, alpha = 0,
                           family = "binomial")

set.seed(9003)
tht_rlr_train <- cv.glmnet(x = train_mat, y = HCMC$thought_do, alpha = 0,
                           family = "binomial")

set.seed(9004)
som_rlr_train <- cv.glmnet(x = train_mat, y = HCMC$som_do, alpha = 0,
                           family = "binomial")

# Test ridge logistic models on external data and consolidate results
rlr_mdls <- list(dep_rlr_train, sub_rlr_train, tht_rlr_train, som_rlr_train)

rlr_out <- matrix(NA,
                  nrow = nrow(rlr_mdls[[1]]$glmnet.fit$beta) + 9,
                  ncol = length(rlr_mdls))
rownames(rlr_out) <- c("Intercept", rownames(rlr_mdls[[1]]$glmnet.fit$beta),
                      "Acc-train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                      "AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(rlr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

test_VAMC_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()
test_PP_mat <- dplyr::select(ppath, one_of(preds)) %>%
  as.matrix()

for(i in 1:length(rlr_mdls)){
  curr_mdl <- rlr_mdls[[i]]
  lambda_idx <- which(curr_mdl$glmnet.fit$lambda == curr_mdl$lambda.min)
  coefs <- curr_mdl$glmnet.fit$beta[, lambda_idx]
  n_coefs <- length(coefs)

  train_pred_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat,
                              type = "response")
  train_pred_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat,
                              type = "class") %>% as.factor()
  test_pred_VAMC_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx =
test_VAMC_mat,
                              type = "response")
  test_pred_VAMC_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx =
test_VAMC_mat,
                              type = "class") %>% as.factor()
  test_pred_PP_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_PP_mat,
                              type = "response")
  test_pred_PP_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_PP_mat,
                              type = "class") %>% as.factor()

  train_obs <- HCMC[, class_outcomes[i]] %>% as.data.frame()
  train_obs <- as.vector(train_obs[, 1]) %>% as.factor()
  test_obs_VAMC <- VAMC[, class_outcomes[i]] %>% as.data.frame()
  test_obs_VAMC <- as.vector(test_obs_VAMC[, 1]) %>% as.factor()
  test_obs_PP <- ppath[, class_outcomes[i]] %>% as.data.frame()
  test_obs_PP <- as.vector(test_obs_PP[, 1]) %>% as.factor()

  seed <- 9000 + i
  set.seed(seed)

```

```

  cv_rslts <- glmnet_cv_class_stats(curr_mdl, x = train_mat, y = HCMC[,
class_outcomes[i]])

  acc_train <- confusionMatrix(train_pred_class, train_obs)$overall['Accuracy']
  acc_cv <- cv_rslts$Acc
  acc_ev_VAMC <- confusionMatrix(test_pred_VAMC_class,
test_obs_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_pred_PP_class, test_obs_PP)$overall['Accuracy']
  AUC_train <- roc(train_obs, train_pred_probs)$auc
  AUC_cv <- cv_rslts$AUC
  AUC_ev_VAMC <- roc(test_obs_VAMC, test_pred_VAMC_probs)$auc
  AUC_ev_PP <- roc(test_obs_PP, test_pred_PP_probs)$auc

  rlr_out[1, i] <- curr_mdl$glmnet.fit$a0[lambda_idx]
  rlr_out[2:(n_coefs + 1), i] <- coefs
  rlr_out[n_coefs + 2, i] <- acc_train
  rlr_out[n_coefs + 3, i] <- acc_cv
  rlr_out[n_coefs + 4, i] <- acc_ev_VAMC
  rlr_out[n_coefs + 5, i] <- acc_ev_PP
  rlr_out[n_coefs + 6, i] <- AUC_train
  rlr_out[n_coefs + 7, i] <- AUC_cv
  rlr_out[n_coefs + 8, i] <- AUC_ev_VAMC
  rlr_out[n_coefs + 9, i] <- AUC_ev_PP
}

setwd(outwd)
write.csv(rlr_out, "Ridge Logistic Regression Output.csv")
setwd(wd)

# Random Forests Classification
#=====

# Get predictor data
preds <- c("rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8",
"rfRC9")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

# Define random forests training control. This will allow the algorithm to explore a
# greater range of tuning parameters and force it to use ROC as its optimizing
function.
rfc_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random",
                           classProbs = TRUE, summaryFunction = twoClassSummary)

# Train RF classification models. Use AUC as performance criterion (preferred over
accuracy
# in this case due to class imbalances)
set.seed(10001)
dep_rfc_train <- train(form = dep_class_form, data = HCMC,
                      trControl = rfc_control, method = "rf",
                      tuneLength = 15, metric = "ROC")

set.seed(10002)
sub_rfc_train <- train(form = sub_class_form, data = HCMC,
                      trControl = rfc_control, method = "rf",
                      tuneLength = 15, metric = "ROC")

set.seed(10003)
tht_rfc_train <- train(form = tht_class_form, data = HCMC,
                      trControl = rfc_control, method = "rf",
                      tuneLength = 15, metric = "ROC")

set.seed(10004)
som_rfc_train <- train(form = som_class_form, data = HCMC,

```

```

        trControl = rfc_control, method = "rf",
        tuneLength = 15, metric = "ROC")

# Export random forests regression results [UPDATE]
rfc_mdls <- list(dep_rfc_train, sub_rfc_train, tht_rfc_train, som_rfc_train)

rfc_out <- matrix(NA,
                 nrow = length(rfc_mdls[[1]]$finalModel$importance) + 7,
                 ncol = length(rfc_mdls))
rownames(rfc_out) <- c(rownames(rfc_mdls[[1]]$finalModel$importance),
                      "Acc-train", "Acc-EV-VAMC", "Acc-EV-PP",
                      "AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(rfc_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(rfc_mdls)){
  curr_mdl <- rfc_mdls[[i]]
  importance <- curr_mdl$finalModel$importance
  n_coefs <- length(importance)

  best_mtry <- curr_mdl$bestTune %>% as.numeric()
  best_idx <- which(curr_mdl$results$mtry == best_mtry)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_mdl$results$ROC[best_idx]
  AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
  AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

  rfc_out[1:n_coefs, i] <- importance
  rfc_out[n_coefs + 1, i] <- acc_train
  rfc_out[n_coefs + 2, i] <- acc_ev_VAMC
  rfc_out[n_coefs + 3, i] <- acc_ev_PP
  rfc_out[n_coefs + 4, i] <- AUC_train
  rfc_out[n_coefs + 5, i] <- AUC_cv
  rfc_out[n_coefs + 6, i] <- AUC_ev_VAMC
  rfc_out[n_coefs + 7, i] <- AUC_ev_PP
}

# Export random forests regression results
setwd(outwd)
write.csv(rfc_out, "Random Forests Classification Output.csv")
setwd(wd)

# Support Vector Machines
#=====

# Train models using full set of training data
set.seed(11001)

```

```

dep_svm_train <- train(form = dep_class_form, data = HCMC,
  preProc = c("center", "scale"),
  trControl = roc_control, method = "svmRadial",
  tuneLength = 14)

set.seed(11002)
sub_svm_train <- train(form = sub_class_form, data = HCMC,
  preProc = c("center", "scale"),
  trControl = roc_control, method = "svmRadial",
  tuneLength = 14)

set.seed(11003)
tht_svm_train <- train(form = tht_class_form, data = HCMC,
  preProc = c("center", "scale"),
  trControl = roc_control, method = "svmRadial",
  tuneLength = 14)

set.seed(11004)
som_svm_train <- train(form = som_class_form, data = HCMC,
  preProc = c("center", "scale"),
  trControl = roc_control, method = "svmRadial",
  tuneLength = 14)

# Evaluate SVM models on external data and consolidate results

svm_mdls <- list(dep_svm_train, sub_svm_train, tht_svm_train, som_svm_train)

svm_out <- matrix(NA,
  nrow = 7,
  ncol = length(svm_mdls))
rownames(svm_out) <- c("Acc-train", "Acc-EV-VAMC", "Acc-EV-PP",
  "AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(svm_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(svm_mdls)){
  curr_mdl <- svm_mdls[[i]]

  best_C <- curr_mdl$bestTune$C
  best_idx <- which(curr_mdl$results$C == best_C)

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_mdl$results$ROC[best_idx]
  AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
  AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

  svm_out[1, i] <- acc_train
  svm_out[2, i] <- acc_ev_VAMC
  svm_out[3, i] <- acc_ev_PP
  svm_out[4, i] <- AUC_train

```



```

    svm_out[5, i] <- AUC_cv
    svm_out[6, i] <- AUC_ev_VAMC
    svm_out[7, i] <- AUC_ev_PP
  }

# Export SVM results
setwd(outwd)
write.csv(svm_out, "Support Vector Machine Output.csv")
setwd(wd)

#=====
# FEATURE SELECTION ANALYSES
#=====

# Set training control parameters
reg_control <- trainControl(method = "cv", number = 10) # For regression
acc_control <- trainControl(method = "cv", number = 10, savePredictions = TRUE, # For
classification accuracy
                             classProbs = TRUE)
roc_control <- trainControl(method = "cv", number = 10, savePredictions = TRUE, # For
classification AUC
                             classProbs = TRUE, summaryFunction = twoClassSummary)

# Set model formulas for repeated use
full_form_preds <- as.character("rfEID + rfBXD + rfTHD + rfRCd + rfRC1 + rfRC2 + rfRC3
+ rfRC4 + rfRC6 + rfRC7 + rfRC8 + rfRC9 +
rfMLS + rfGIC + rfHPC + rfNUC + rfCOG + rfSUI + rfHLP + rfSFD + rfNFC + rfSTW + rfAXY
+ rfANP + rfBRF + rfMSF +
rfJCP + rfSUB + rfAGG + rfACT + rfFML + rfIPP + rfSAV + rfSHY + rfDSF + rfAES + rfMEC
+ rfAGGR + rfPSYC + rfDISC +
rfNEGE + rfINTR")

full_dep_reg_form <- paste("msefct1 ~ ", full_form_preds) %>% as.formula()
full_sub_reg_form <- paste("msefct4 ~ ", full_form_preds) %>% as.formula()
full_tht_reg_form <- paste("msefct5 ~ ", full_form_preds) %>% as.formula()
full_pai_reg_form <- paste("msefct8 ~ ", full_form_preds) %>% as.formula()

full_dep_class_form <- paste("dep_do ~ ", full_form_preds) %>% as.formula()
full_sub_class_form <- paste("sub_do ~ ", full_form_preds) %>% as.formula()
full_tht_class_form <- paste("thought_do ~ ", full_form_preds) %>% as.formula()
full_som_class_form <- paste("som_do ~ ", full_form_preds) %>% as.formula()

# Bivariate Linear Regression
#=====

# Check bivariate correlations between predictors and criteria in training data
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
"rfRC6", "rfRC7", "rfRC8", "rfRC9",
          "rfMLS", "rfGIC", "rfHPC", "rfNUC", "rfCOG", "rfSUI", "rfHLP", "rfSFD",
"rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF", "rfMSF",
          "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
"rfDSF", "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC",
          "rfNEGE", "rfINTR")
criteria <- c("msefct1", "msefct4", "msefct5", "msefct8")
pred_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()
crit_mat <- dplyr::select(HCMC, one_of(criteria))
reg_corrs <- cor(pred_mat, crit_mat, method = "pearson")

# Train models using training data set. Each model has one predictor: the RF scale
with which

```

```

# the criterion is most strongly correlated at the zero-order level.
set.seed(5001)
dep_bi_train <- train(form = msefct1 ~ rfEID, data = HCMC,
                      trControl = cv_control, method = "lm")
set.seed(5002)
sub_bi_train <- train(form = msefct4 ~ rfSUB, data = HCMC,
                      trControl = cv_control, method = "lm")
set.seed(5003)
tht_bi_train <- train(form = msefct5 ~ rfSFD, data = HCMC,
                      trControl = cv_control, method = "lm")
set.seed(5004)
pai_bi_train <- train(form = msefct8 ~ rfHPC, data = HCMC,
                      trControl = cv_control, method = "lm")

# Export bivariate regression results
bi_mdls <- list(dep_bi_train, sub_bi_train, tht_bi_train, pai_bi_train)

bi_out <- matrix(NA, nrow = 9, ncol = length(lm_mdls))

rownames(bi_out) <- c("Intercept", "B", "R2-train",
                     "Adj R2 train", "RMSE-train", "R2-CV", "RMSE-CV",
                     "R2-EV", "RMSE-EV")
colnames(bi_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(bi_mdls)){
  curr_mdl <- bi_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- summary(curr_mdl)$r.squared
  adj_R2_train <- summary(curr_mdl)$adj.r.squared
  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  R2_cv <- curr_mdl$results$Rsquared
  rmse_cv <- curr_mdl$results$RMSE
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  bi_out[1:n_coefs, i] <- coefs
  bi_out[n_coefs + 1, i] <- R2_train
  bi_out[n_coefs + 2, i] <- adj_R2_train
  bi_out[n_coefs + 3, i] <- rmse_train
  bi_out[n_coefs + 4, i] <- R2_cv
  bi_out[n_coefs + 5, i] <- rmse_cv
  bi_out[n_coefs + 6, i] <- R2_ev
  bi_out[n_coefs + 7, i] <- rmse_ev
}

setwd(outwd)
write.csv(bi_out, "Bivariate Regression Output.csv")
setwd(wd)

# Bivariate logistic regression
#=====

# Compute AUCs for RC scales against diagnostic group membership variables in
# training data.

```

```

dep_auc <- rf_auc(HCMC, "dep_do") # best = RC2, AUC = .74
sub_auc <- rf_auc(HCMC, "sub_do") # best = SUB, AUC = .79
tht_auc <- rf_auc(HCMC, "thought_do") # best = EID, AUC = .71
som_auc <- rf_auc(HCMC, "som_do") # best = MEC, AUC = .88

# Train bivariate logist regression models using full training data
set.seed(8001)
dep_blr_train <- train(form = dep_do ~ rfRC2, data = HCMC,
                      trControl = acc_control, method = "glm")

set.seed(8002)
sub_blr_train <- train(form = sub_do ~ rfSUB, data = HCMC,
                      trControl = acc_control, method = "glm")

set.seed(8003)
tht_blr_train <- train(form = thought_do ~ rfEID, data = HCMC,
                      trControl = acc_control, method = "glm")

set.seed(8004)
som_blr_train <- train(form = som_do ~ rfMEC, data = HCMC,
                      trControl = acc_control, method = "glm")

# Recompute models using to obtain cross-validate ROC AUC (caret does not
# simultaneously calculate Accuracy and ROC)
set.seed(8001)
dep_blr_roc <- train(form = dep_do ~ rfRCd, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")

set.seed(8002)
sub_blr_roc <- train(form = sub_do ~ rfRC4, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")

set.seed(8003)
tht_blr_roc <- train(form = thought_do ~ rfRCd, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")

set.seed(8004)
som_blr_roc <- train(form = som_do ~ rfRC9, data = HCMC,
                    trControl = roc_control, method = "glm",
                    metric = "ROC")

# Test bivariate models on external data and consolidate results
blr_acc_mdls <- list(dep_blr_train, sub_blr_train, tht_blr_train, som_blr_train)
blr_roc_mdls <- list(dep_blr_roc, sub_blr_roc, tht_blr_roc, som_blr_roc)

blr_out <- matrix(NA,
                 nrow = length(blr_acc_mdls[[1]]$finalModel$coefficients) + 8,
                 ncol = length(blr_acc_mdls))
rownames(blr_out) <- c(names(blr_acc_mdls[[1]]$finalModel$coefficients),
                      "Acc-Train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                      "AUC-Train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(blr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(blr_acc_mdls)){
  curr_mdl <- blr_acc_mdls[[i]]
  curr_roc <- blr_roc_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)

```

```

test_PP <- predict.train(curr_mdl, ppath)
test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
acc_cv <- curr_mdl$results$Accuracy
acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
AUC_train <- roc(train_outcome, train_probs$Pos)$auc
AUC_cv <- curr_mdl$results$ROC
AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

blr_out[1:n_coefs, i] <- coefs
blr_out[n_coefs + 1, i] <- acc_train
blr_out[n_coefs + 2, i] <- acc_cv
blr_out[n_coefs + 3, i] <- acc_ev_VAMC
blr_out[n_coefs + 4, i] <- acc_ev_PP
blr_out[n_coefs + 5, i] <- AUC_train
blr_out[n_coefs + 6, i] <- AUC_cv
blr_out[n_coefs + 7, i] <- AUC_ev_VAMC
blr_out[n_coefs + 8, i] <- AUC_ev_PP
}

# Export bivariate logistic regression results
setwd(outwd)
write.csv(bl_r_out, "Bivariate Logistic Regression Output.csv")
setwd(wd)

# Full-Scale OLS Linear Regression
#=====

# Train models using full training data
set.seed(12001)
full_dep_reg_lm <- train(form = full_dep_reg_form, data = HCMC,
                        trControl = reg_control, method = "lm")
set.seed(12002)
full_sub_reg_lm <- train(form = full_sub_reg_form, data = HCMC,
                        trControl = reg_control, method = "lm")
set.seed(12003)
full_tht_reg_lm <- train(form = full_tht_reg_form, data = HCMC,
                        trControl = reg_control, method = "lm")
set.seed(12004)
full_pai_reg_lm <- train(form = full_pai_reg_form, data = HCMC,
                        trControl = reg_control, method = "lm")

# Test full-scale OLS regression models on external data and consolidate results
full_lm_mdls <- list(full_dep_reg_lm, full_sub_reg_lm, full_tht_reg_lm,
                    full_pai_reg_lm)

full_lm_out <- matrix(NA,
                    nrow = length(full_lm_mdls[[1]]$finalModel$coefficients) + 7,
                    ncol = length(full_lm_mdls))
rownames(full_lm_out) <- c(names(full_lm_mdls[[1]]$finalModel$coefficients), "R2-
train",
                        "Adj R2 train", "RMSE-train", "R2-CV", "RMSE-CV",
                        "R2-EV", "RMSE-EV")
colnames(full_lm_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

```

```

for(i in 1:length(full_lm_mdls)){
  curr_mdl <- full_lm_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- summary(curr_mdl)$r.squared
  adj_R2_train <- summary(curr_mdl)$adj.r.squared
  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  R2_cv <- curr_mdl$results$Rsquared
  rmse_cv <- curr_mdl$results$RMSE
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  full_lm_out[1:n_coefs, i] <- coefs
  full_lm_out[n_coefs + 1, i] <- R2_train
  full_lm_out[n_coefs + 2, i] <- adj_R2_train
  full_lm_out[n_coefs + 3, i] <- rmse_train
  full_lm_out[n_coefs + 4, i] <- R2_cv
  full_lm_out[n_coefs + 5, i] <- rmse_cv
  full_lm_out[n_coefs + 6, i] <- R2_ev
  full_lm_out[n_coefs + 7, i] <- rmse_ev
}

# Export full-scale OLS linear regression results
setwd(outwd)
write.csv(full_lm_out, "Full-Scale OLS Linear Regression Output.csv")
setwd(wd)

# Full-Scale Logistic Regression
#=====

# Train models using full training data
set.seed(13001)
full_dep_class_lr <- train(form = full_dep_class_form, data = HCMC,
                           trControl = acc_control, method = "glm")
set.seed(13002)
full_sub_class_lr <- train(form = full_sub_class_form, data = HCMC,
                           trControl = acc_control, method = "glm")
set.seed(13003)
full_tht_class_lr <- train(form = full_tht_class_form, data = HCMC,
                           trControl = acc_control, method = "glm")
set.seed(13004)
full_som_class_lr <- train(form = full_som_class_form, data = HCMC,
                           trControl = acc_control, method = "glm")

# Recompute models using to obtain cross-validate ROC AUC (caret does not
# simultaneously calculate Accuracy and ROC)
set.seed(13001)
full_dep_roc_lr <- train(form = full_dep_class_form, data = HCMC,
                         trControl = roc_control, method = "glm")
set.seed(13002)
full_sub_roc_lr <- train(form = full_sub_class_form, data = HCMC,
                         trControl = roc_control, method = "glm")
set.seed(13003)
full_tht_roc_lr <- train(form = full_tht_class_form, data = HCMC,
                         trControl = roc_control, method = "glm")
set.seed(13004)
full_som_roc_lr <- train(form = full_som_class_form, data = HCMC,

```

```

trControl = roc_control, method = "glm")

# Test models on external data and consolidate results
full_lr_acc_mdls <- list(full_dep_class_lr, full_sub_class_lr, full_tht_class_lr,
full_som_class_lr)
full_lr_roc_mdls <- list(dep_lr_roc, sub_lr_roc, tht_lr_roc, som_lr_roc)

full_lr_out <- matrix(NA,
                     nrow = length(full_lr_acc_mdls[[1]]$finalModel$coefficients) + 8,
                     ncol = length(full_lr_acc_mdls))
rownames(full_lr_out) <- c(names(full_lr_acc_mdls[[1]]$finalModel$coefficients),
                          "Acc-Train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                          "AUC-Train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(full_lr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(full_lr_acc_mdls)){
  curr_mdl <- full_lr_acc_mdls[[i]]
  curr_roc <- full_lr_roc_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  n_coefs <- length(coefs)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_cv <- curr_mdl$results$Accuracy
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_roc$results$ROC
  AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
  AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

  full_lr_out[1:n_coefs, i] <- coefs
  full_lr_out[n_coefs + 1, i] <- acc_train
  full_lr_out[n_coefs + 2, i] <- acc_cv
  full_lr_out[n_coefs + 3, i] <- acc_ev_VAMC
  full_lr_out[n_coefs + 4, i] <- acc_ev_PP
  full_lr_out[n_coefs + 5, i] <- AUC_train
  full_lr_out[n_coefs + 6, i] <- AUC_cv
  full_lr_out[n_coefs + 7, i] <- AUC_ev_VAMC
  full_lr_out[n_coefs + 8, i] <- AUC_ev_PP
}

# Export logistic regression results
setwd(outwd)
write.csv(full_lr_out, "Full-Scale Logistic Regression Output.csv")
setwd(wd)

# Full-Scale Ridge Regression using glmnet
#=====

```

```

# Train models using full training data
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
           "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
           "rfCOG",
           "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
           "rfMSF",
           "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
           "rfDSF",
           "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

set.seed(14001)
full_dep_ridge_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct1, alpha = 0)
set.seed(14002)
full_sub_ridge_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct4, alpha = 0)
set.seed(14003)
full_tht_ridge_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct5, alpha = 0)
set.seed(14004)
full_pai_ridge_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct8, alpha = 0)

# Export ridge regression results
full_ridge_mdls <- list(full_dep_ridge_reg, full_sub_ridge_reg, full_tht_ridge_reg,
full_pai_ridge_reg)

full_ridge_out <- matrix(NA,
                        nrow = nrow(full_ridge_mdls[[1]]$glmnet.fit$beta) + 7,
                        ncol = length(full_ridge_mdls))
rownames(full_ridge_out) <- c("Intercept",
rownames(full_ridge_mdls[[1]]$glmnet.fit$beta),
                        "R2-train", "R2-CV", "R2-EV",
                        "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(full_ridge_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

test_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()

for(i in 1:length(full_ridge_mdls)){
  curr_mdl <- full_ridge_mdls[[i]]
  lambda_idx <- which(curr_mdl$glmnet.fit$lambda == curr_mdl$lambda.min)
  coefs <- curr_mdl$glmnet.fit$beta[, lambda_idx]
  n_coefs <- length(coefs)

  train_outcome <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat)
  test_pred <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_mat)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- R2(train_outcome, HCMC[, reg_outcomes[i]])
  R2_cv <- glmnet_cv_R2(curr_mdl, x = train_mat, y = as.matrix(HCMC[,
reg_outcomes[i]]))
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_train <- RMSE(train_outcome, HCMC[, reg_outcomes[i]])
  rmse_cv <- glmnet_cv_rmse(curr_mdl)
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  full_ridge_out[1, i] <- curr_mdl$glmnet.fit$a0[lambda_idx]
  full_ridge_out[2:(n_coefs + 1), i] <- coefs
  full_ridge_out[n_coefs + 2, i] <- R2_train
  full_ridge_out[n_coefs + 3, i] <- R2_cv
  full_ridge_out[n_coefs + 4, i] <- R2_ev
  full_ridge_out[n_coefs + 5, i] <- rmse_train

```

```

    full_ridge_out[n_coefs + 6, i] <- rmse_cv
    full_ridge_out[n_coefs + 7, i] <- rmse_ev
  }

  setwd(outwd)
  write.csv(full_ridge_out, "Full-Scale Ridge Regression Output.csv")
  setwd(wd)

# Full-Scale Ridge Logistic Regression using glmnet
#=====

# Train models using full training data
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
           "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
           "rfCOG",
           "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
           "rfMSF",
           "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
           "rfDSF",
           "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

set.seed(15001)
full_dep_rlr_class <- cv.glmnet(x = train_mat, y = HCMC$dep_do, alpha = 0,
                               family = "binomial")

set.seed(15002)
full_sub_rlr_class <- cv.glmnet(x = train_mat, y = HCMC$sub_do, alpha = 0,
                               family = "binomial")

set.seed(15003)
full_tht_rlr_class <- cv.glmnet(x = train_mat, y = HCMC$thought_do, alpha = 0,
                               family = "binomial")

set.seed(15004)
full_som_rlr_class <- cv.glmnet(x = train_mat, y = HCMC$som_do, alpha = 0,
                               family = "binomial")

# Test ridge logistic models on external data and consolidate results
full_rlr_mdls <- list(full_dep_rlr_class, full_sub_rlr_class, full_tht_rlr_class,
                      full_som_rlr_class)

full_rlr_out <- matrix(NA,
                      nrow = nrow(full_rlr_mdls[[1]]$glmnet.fit$beta) + 9,
                      ncol = length(full_rlr_mdls))
rownames(full_rlr_out) <- c("Intercept", rownames(full_rlr_mdls[[1]]$glmnet.fit$beta),
                           "Acc-train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                           "AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(full_rlr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

test_VAMC_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()
test_PP_mat <- dplyr::select(ppath, one_of(preds)) %>%
  as.matrix()

for(i in 1:length(rlr_mdls)){
  curr_mdl <- full_rlr_mdls[[i]]
  lambda_idx <- which(curr_mdl$glmnet.fit$lambda == curr_mdl$lambda.min)
  coefs <- curr_mdl$glmnet.fit$beta[, lambda_idx]
  n_coefs <- length(coefs)

  train_pred_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat,

```



```

                                type = "response")
train_pred_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat,
                                type = "class") %>% as.factor()
test_pred_VAMC_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx =
test_VAMC_mat,
                                type = "response")
test_pred_VAMC_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx =
test_VAMC_mat,
                                type = "class") %>% as.factor()
test_pred_PP_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_PP_mat,
                                type = "response")
test_pred_PP_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_PP_mat,
                                type = "class") %>% as.factor()

train_obs <- HCMC[, class_outcomes[i]] %>% as.data.frame()
train_obs <- as.vector(train_obs[, 1]) %>% as.factor()
test_obs_VAMC <- VAMC[, class_outcomes[i]] %>% as.data.frame()
test_obs_VAMC <- as.vector(test_obs_VAMC[,1]) %>% as.factor()
test_obs_PP <- ppath[, class_outcomes[i]] %>% as.data.frame()
test_obs_PP <- as.vector(test_obs_PP[,1]) %>% as.factor()

seed <- 9000 + i
set.seed(seed)
cv_rslts <- glmnet_cv_class_stats(curr_mdl, x = train_mat, y = HCMC[,
class_outcomes[i]])

acc_train <- confusionMatrix(train_pred_class, train_obs)$overall['Accuracy']
acc_cv <- cv_rslts$Acc
acc_ev_VAMC <- confusionMatrix(test_pred_VAMC_class,
test_obs_VAMC)$overall['Accuracy']
acc_ev_PP <- confusionMatrix(test_pred_PP_class, test_obs_PP)$overall['Accuracy']
AUC_train <- roc(train_obs, train_pred_probs)$auc
AUC_cv <- cv_rslts$AUC
AUC_ev_VAMC <- roc(test_obs_VAMC, test_pred_VAMC_probs)$auc
AUC_ev_PP <- roc(test_obs_PP, test_pred_PP_probs)$auc

full_rlr_out[1, i] <- curr_mdl$glmnet.fit$a0[lambda_idx]
full_rlr_out[2:(n_coefs + 1), i] <- coefs
full_rlr_out[n_coefs + 2, i] <- acc_train
full_rlr_out[n_coefs + 3, i] <- acc_cv
full_rlr_out[n_coefs + 4, i] <- acc_ev_VAMC
full_rlr_out[n_coefs + 5, i] <- acc_ev_PP
full_rlr_out[n_coefs + 6, i] <- AUC_train
full_rlr_out[n_coefs + 7, i] <- AUC_cv
full_rlr_out[n_coefs + 8, i] <- AUC_ev_VAMC
full_rlr_out[n_coefs + 9, i] <- AUC_ev_PP
}

setwd(outwd)
write.csv(full_rlr_out, "Full-Scale Ridge Logistic Regression Output.csv")
setwd(wd)

# Manual Selection Linear Regression
#=====

# Train models using full training data
set.seed(16001)
man_dep_reg_lm <- train(form = msefct1 ~ rfRCd + rfRC2 + rfMLS, data = HCMC,
                        trControl = reg_control, method = "lm")
set.seed(16002)
man_sub_reg_lm <- train(form = msefct4 ~ rfSUB + rfDISC, data = HCMC,
                        trControl = reg_control, method = "lm")

```

```

set.seed(16003)
man_tht_reg_lm <- train(form = msefct5 ~ rfRC6 + rfRC8, data = HCMC,
                        trControl = reg_control, method = "lm")
set.seed(16004)
man_pai_reg_lm <- train(form = msefct8 ~ rfMLS + rfGIC + rfHPC, data = HCMC,
                        trControl = reg_control, method = "lm")

# Test manually selected regression models on external data and consolidate results

coef_nams <- c("(Intercept)", "rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2",
               "rfRC3", "rfRC4",
               "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
               "rfCOG",
               "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
               "rfMSF",
               "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
               "rfDSF",
               "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")

man_lm_mdls <- list(man_dep_reg_lm, man_sub_reg_lm, man_tht_reg_lm, man_pai_reg_lm)

man_lm_out <- matrix(NA,
                    nrow = length(coef_nams) + 7,
                    ncol = length(man_lm_mdls))
rownames(man_lm_out) <- c(coef_nams, "R2-train",
                          "Adj R2 train", "RMSE-train", "R2-CV", "RMSE-CV",
                          "R2-EV", "RMSE-EV")
colnames(man_lm_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(man_lm_mdls)){
  curr_mdl <- man_lm_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  coef_idx <- which(coef_nams %in% names(coefs))
  n_coefs <- length(coef_nams)

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- summary(curr_mdl)$r.squared
  adj_R2_train <- summary(curr_mdl)$adj.r.squared
  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  R2_cv <- curr_mdl$results$Rsquared
  rmse_cv <- curr_mdl$results$RMSE
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  man_lm_out[coef_idx, i] <- coefs
  man_lm_out[n_coefs + 1, i] <- R2_train
  man_lm_out[n_coefs + 2, i] <- adj_R2_train
  man_lm_out[n_coefs + 3, i] <- rmse_train
  man_lm_out[n_coefs + 4, i] <- R2_cv
  man_lm_out[n_coefs + 5, i] <- rmse_cv
  man_lm_out[n_coefs + 6, i] <- R2_ev
  man_lm_out[n_coefs + 7, i] <- rmse_ev
}

# Export manual selection linear regression results
setwd(outwd)
write.csv(man_lm_out, "Manual Selection Linear Regression Output.csv")

```

```

setwd(wd)

# Manual Selection Logistic Regression
#=====

# Train models using full training data
set.seed(17001)
man_dep_class_lr <- train(form = dep_do ~ rfRCd + rfRC2 + rfMLS, data = HCMC,
                          trControl = acc_control, method = "glm")
set.seed(17002)
man_sub_class_lr <- train(form = sub_do ~ rfSUB + rfDISC, data = HCMC,
                          trControl = acc_control, method = "glm")
set.seed(17003)
man_tht_class_lr <- train(form = thought_do ~ rfRC6 + rfRC8, data = HCMC,
                          trControl = acc_control, method = "glm")
set.seed(17004)
man_som_class_lr <- train(form = som_do ~ rfRC1 + rfMEC, data = HCMC,
                          trControl = acc_control, method = "glm")

# Recompute models using to obtain cross-validate ROC AUC (caret does not
# simultaneously calculate Accuracy and ROC)
set.seed(17001)
man_dep_roc_lr <- train(form = dep_do ~ rfRCd + rfRC2 + rfMLS, data = HCMC,
                        trControl = roc_control, method = "glm")
set.seed(17002)
man_sub_roc_lr <- train(form = sub_do ~ rfSUB + rfDISC, data = HCMC,
                        trControl = roc_control, method = "glm")
set.seed(17003)
man_tht_roc_lr <- train(form = thought_do ~ rfRC6 + rfRC8, data = HCMC,
                        trControl = roc_control, method = "glm")
set.seed(17004)
man_som_roc_lr <- train(form = som_do ~ rfRC1 + rfMEC, data = HCMC,
                        trControl = roc_control, method = "glm")

# Test models on external data and consolidate results
coef_nams <- c("(Intercept)", "rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2",
"rfRC3", "rfRC4",
               "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
"rfCOG",
               "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
"rfMSF",
               "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
"rfDSF",
               "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")

man_lr_acc_mdls <- list(man_dep_class_lr, man_sub_class_lr, man_tht_class_lr,
man_som_class_lr)
man_lr_roc_mdls <- list(dep_lr_roc, sub_lr_roc, tht_lr_roc, som_lr_roc)

man_lr_out <- matrix(NA,
                    nrow = length(coef_nams) + 8,
                    ncol = length(man_lr_acc_mdls))
rownames(man_lr_out) <- c(coef_nams,
                          "Acc-Train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                          "AUC-Train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(man_lr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(man_lr_acc_mdls)){
  curr_md1 <- man_lr_acc_mdls[[i]]

```

```

curr_roc <- man_lr_roc_mdls[[i]]
coefs <- curr_mdl$finalModel$coefficients
coef_idx <- which(coef_nams %in% names(coefs))
n_coefs <- length(coef_nams)

train_outcome <- curr_mdl$trainingData$outcome
train_pred <- predict.train(curr_mdl, HCMC)
train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
test_VAMC <- predict.train(curr_mdl, VAMC)
test_PP <- predict.train(curr_mdl, ppath)
test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
acc_cv <- curr_mdl$results$Accuracy
acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
AUC_train <- roc(train_outcome, train_probs$Pos)$auc
AUC_cv <- curr_roc$results$ROC
AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

man_lr_out[coef_idx, i] <- coefs
man_lr_out[n_coefs + 1, i] <- acc_train
man_lr_out[n_coefs + 2, i] <- acc_cv
man_lr_out[n_coefs + 3, i] <- acc_ev_VAMC
man_lr_out[n_coefs + 4, i] <- acc_ev_PP
man_lr_out[n_coefs + 5, i] <- AUC_train
man_lr_out[n_coefs + 6, i] <- AUC_cv
man_lr_out[n_coefs + 7, i] <- AUC_ev_VAMC
man_lr_out[n_coefs + 8, i] <- AUC_ev_PP
}

# Export logistic regression results
setwd(outwd)
write.csv(man_lr_out, "Manual Selection Logistic Regression Output.csv")
setwd(wd)

# Stepwise Selection Linear Regression
#=====

# Note that this section includes training and testing of three different types of
# stepwise models (forward, backward, and forward-backward hybrid).

# Train forward stepwise models using full training data
set.seed(18001)
dep_reg_fsw <- train(form = full_dep_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE)

set.seed(18002)
sub_reg_fsw <- train(form = full_sub_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE)

set.seed(18003)
tht_reg_fsw <- train(form = full_tht_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE)

set.seed(18004)
pai_reg_fsw <- train(form = full_pai_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",

```

```

        direction = "forward", trace = FALSE)

# Train backward stepwise models using full training data
set.seed(19001)
dep_reg_bsw <- train(form = full_dep_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "backward", trace = FALSE)

set.seed(19002)
sub_reg_bsw <- train(form = full_sub_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "backward", trace = FALSE)

set.seed(19003)
tht_reg_bsw <- train(form = full_tht_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "backward", trace = FALSE)

set.seed(19004)
pai_reg_bsw <- train(form = full_pai_reg_form, data = HCMC,
                     trControl = reg_control, method = "glmStepAIC",
                     direction = "backward", trace = FALSE)

# Train hybrid (forward and backward) stepwise models using full training data

# intercept models for forward-backward selection
dep_reg_int <- lm(msefct1 ~ 1, data = HCMC)
sub_reg_int <- lm(msefct4 ~ 1, data = HCMC)
tht_reg_int <- lm(msefct5 ~ 1, data = HCMC)
pai_reg_int <- lm(msefct8 ~ 1, data = HCMC)

# full models for forward-backward selection
dep_reg_full <- lm(full_dep_reg_form, data = HCMC)
sub_reg_full <- lm(full_sub_reg_form, data = HCMC)
tht_reg_full <- lm(full_tht_reg_form, data = HCMC)
pai_reg_full <- lm(full_pai_reg_form, data = HCMC)

set.seed(20001)
dep_reg_hsw <- stepAIC(object = dep_reg_int,
                      scope = list(upper = dep_reg_full, lower = dep_reg_int),
                      direction = "both", trace = FALSE)

set.seed(20002)
sub_reg_hsw <- stepAIC(object = sub_reg_int,
                      scope = list(upper = sub_reg_full, lower = sub_reg_int),
                      direction = "both", trace = FALSE)

set.seed(20003)
tht_reg_hsw <- stepAIC(object = tht_reg_int,
                      scope = list(upper = tht_reg_full, lower = tht_reg_int),
                      direction = "both", trace = FALSE)

set.seed(20004)
pai_reg_hsw <- stepAIC(object = pai_reg_int,
                      scope = list(upper = pai_reg_full, lower = pai_reg_int),
                      direction = "both", trace = FALSE)

# Test stepwise linear regression models on external data and consolidate results
coef_nams <- c("(Intercept)", "rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2",
               "rfRC3", "rfRC4",
               "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
               "rfCOG",
               "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
               "rfMSF",
               "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
               "rfDSF",
               "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")

```

```

fsw_reg_mdls <- list(dep_reg_fsw, sub_reg_fsw, tht_reg_fsw, pai_reg_fsw)
bsw_reg_mdls <- list(dep_reg_bsw, sub_reg_bsw, tht_reg_bsw, pai_reg_bsw)
hsw_reg_mdls <- list(dep_reg_hsw, sub_reg_hsw, tht_reg_hsw, pai_reg_hsw)

fsw_reg_out <- matrix(NA,
                      nrow = length(coef_nams) + 6,
                      ncol = length(fsw_reg_mdls))

rownames(fsw_reg_out) <- c(coef_nams, "R2-train", "R2-CV", "R2-EV",
                           "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(fsw_reg_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

bsw_reg_out <- hsw_reg_out <- fsw_reg_out

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

# Consolidate forward stepwise regression results
for(i in 1:length(fsw_reg_mdls)){
  curr_mdl <- fsw_reg_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  coef_idx <- which(coef_nams %in% names(coefs))
  n_coefs <- length(coef_nams)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- R2(train_pred, train_outcome)
  adj_R2_train <- summary(curr_mdl)$adj.r.squared
  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  R2_cv <- curr_mdl$results$Rsquared
  rmse_cv <- curr_mdl$results$RMSE
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  # Place coefficients in correct cells
  for(j in 1:length(coefs)){
    par_nams <- names(coefs)
    coef_idx <- which(rownames(fsw_reg_out) == par_nams[j])
    fsw_reg_out[coef_idx, i] <- coefs[j]
  }

  fsw_reg_out[n_coefs + 1, i] <- R2_train
  fsw_reg_out[n_coefs + 2, i] <- R2_cv
  fsw_reg_out[n_coefs + 3, i] <- R2_ev
  fsw_reg_out[n_coefs + 4, i] <- rmse_train
  fsw_reg_out[n_coefs + 5, i] <- rmse_cv
  fsw_reg_out[n_coefs + 6, i] <- rmse_ev
}

# Consolidate backward stepwise regression results
for(i in 1:length(bsw_reg_mdls)){
  curr_mdl <- bsw_reg_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  coef_idx <- which(coef_nams %in% names(coefs))
  n_coefs <- length(coef_nams)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

```

```

R2_train <- R2(train_pred, train_outcome)
adj_R2_train <- summary(curr_mdl)$adj.r.squared
rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
R2_cv <- curr_mdl$results$Rsquared
rmse_cv <- curr_mdl$results$RMSE
R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

# Place coefficients in correct cells
for(j in 1:length(coefs)){
  par_nams <- names(coefs)
  coef_idx <- which(rownames(bsw_reg_out) == par_nams[j])
  bsw_reg_out[coef_idx, i] <- coefs[j]
}

bsw_reg_out[n_coefs + 1, i] <- R2_train
bsw_reg_out[n_coefs + 2, i] <- R2_cv
bsw_reg_out[n_coefs + 3, i] <- R2_ev
bsw_reg_out[n_coefs + 4, i] <- rmse_train
bsw_reg_out[n_coefs + 5, i] <- rmse_cv
bsw_reg_out[n_coefs + 6, i] <- rmse_ev
}

# Consolidate hybrid stepwise regression results
hsw_reg_ints <- list(dep_reg_int, sub_reg_int, tht_reg_int, pai_reg_int)
hsw_reg_fulls <- list(dep_reg_full, sub_reg_full, tht_reg_full, pai_reg_full)
for(i in 1:length(hsw_reg_mdls)){
  curr_mdl <- hsw_reg_mdls[[i]]
  curr_int <- hsw_reg_ints[[i]]
  curr_full <- hsw_reg_fulls[[i]]
  coefs <- curr_mdl$coefficients
  n_coefs <- length(coef_nams)

  train_outcome <- curr_mdl$model[1]
  train_pred <- predict(curr_mdl, HCMC)
  test_pred <- predict(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- R2(train_pred, train_outcome)
  adj_R2_train <- summary(curr_mdl)$adj.r.squared
  rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  # Get cross-validated performance
  cv_rslts <- step_cv_reg(curr_int, curr_full, data = HCMC)
  R2_cv <- cv_rslts$R2
  rmse_cv <- cv_rslts$RMSE

  # Place coefficients in correct cells
  for(j in 1:length(coefs)){
    par_nams <- names(coefs)
    coef_idx <- which(rownames(hsw_reg_out) == par_nams[j])
    hsw_reg_out[coef_idx, i] <- coefs[j]
  }

  hsw_reg_out[n_coefs + 1, i] <- R2_train
  hsw_reg_out[n_coefs + 2, i] <- R2_cv
  hsw_reg_out[n_coefs + 3, i] <- R2_ev
  hsw_reg_out[n_coefs + 4, i] <- rmse_train
  hsw_reg_out[n_coefs + 5, i] <- rmse_cv
  hsw_reg_out[n_coefs + 6, i] <- rmse_ev
}

```

```

# Output stepwise regression results.
setwd(outwd)
write.csv(fsw_reg_out, "Forward Stepwise Regression Output.csv")
write.csv(bsw_reg_out, "Backward Stepwise Regression Output.csv")
write.csv(hsw_reg_out, "Hybrid Stepwise Regression Output.csv")
setwd(wd)

# Stepwise Selection Logistic Regression
=====

# Note that this section includes training and testing of three different types of
# stepwise models (forward, backward, and forward-backward hybrid).

# Train forward stepwise logistic regression models using full training data
set.seed(21001)
dep_class_fsw <- train(form = full_dep_class_form, data = HCMC,
                       trControl = acc_control, method = "glmStepAIC",
                       direction = "forward", trace = FALSE, family = "binomial")

set.seed(21002)
sub_class_fsw <- train(form = full_sub_class_form, data = HCMC,
                       trControl = acc_control, method = "glmStepAIC",
                       direction = "forward", trace = FALSE, family = "binomial")

set.seed(21003)
tth_class_fsw <- train(form = full_tth_class_form, data = HCMC,
                       trControl = acc_control, method = "glmStepAIC",
                       direction = "forward", trace = FALSE, family = "binomial")

set.seed(21004)
som_class_fsw <- train(form = full_som_class_form, data = HCMC,
                       trControl = acc_control, method = "glmStepAIC",
                       direction = "forward", trace = FALSE, family = "binomial")

# Recompute forward stepwise logistic regression models to obtain cross-validate ROC
# AUC
set.seed(21001)
dep_roc_fsw <- train(form = full_dep_class_form, data = HCMC,
                     trControl = roc_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE, family = "binomial")

set.seed(21002)
sub_roc_fsw <- train(form = full_sub_class_form, data = HCMC,
                     trControl = roc_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE, family = "binomial")

set.seed(21003)
tth_roc_fsw <- train(form = full_tth_class_form, data = HCMC,
                     trControl = roc_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE, family = "binomial")

set.seed(21004)
som_roc_fsw <- train(form = full_som_class_form, data = HCMC,
                     trControl = roc_control, method = "glmStepAIC",
                     direction = "forward", trace = FALSE, family = "binomial")

# Train backward stepwise logistic regression models using full training data
set.seed(22001)
dep_class_bsw <- train(form = full_dep_class_form, data = HCMC,
                       trControl = acc_control, method = "glmStepAIC",
                       direction = "backward", trace = FALSE, family = "binomial")

set.seed(22002)
sub_class_bsw <- train(form = full_sub_class_form, data = HCMC,
                       trControl = acc_control, method = "glmStepAIC",
                       direction = "backward", trace = FALSE, family = "binomial")

set.seed(22003)

```



```

tth_class_bsw <- train(form = full_tth_class_form, data = HCMC,
                      trControl = acc_control, method = "glmStepAIC",
                      direction = "backward", trace = FALSE, family = "binomial")
set.seed(22004)
som_class_bsw <- train(form = full_som_class_form, data = HCMC,
                      trControl = acc_control, method = "glmStepAIC",
                      direction = "backward", trace = FALSE, family = "binomial")

# Recompute backward stepwise logistic regression models to obtain cross-validated ROC
# AUC
set.seed(22001)
dep_roc_bsw <- train(form = full_dep_class_form, data = HCMC,
                    trControl = roc_control, method = "glmStepAIC",
                    direction = "backward", trace = FALSE, family = "binomial")
set.seed(22002)
sub_roc_bsw <- train(form = full_sub_class_form, data = HCMC,
                    trControl = roc_control, method = "glmStepAIC",
                    direction = "backward", trace = FALSE, family = "binomial")
set.seed(22003)
tth_roc_bsw <- train(form = full_tth_class_form, data = HCMC,
                    trControl = roc_control, method = "glmStepAIC",
                    direction = "backward", trace = FALSE, family = "binomial")
set.seed(22004)
som_roc_bsw <- train(form = full_som_class_form, data = HCMC,
                    trControl = roc_control, method = "glmStepAIC",
                    direction = "backward", trace = FALSE, family = "binomial")

# Train hybrid stepwise logistic regression models using full training data

# intercept models for forward-backward selection
dep_class_int <- glm(dep_do ~ 1, data = HCMC, family = "binomial")
sub_class_int <- glm(sub_do ~ 1, data = HCMC, family = "binomial")
tth_class_int <- glm(thought_do ~ 1, data = HCMC, family = "binomial")
som_class_int <- glm(som_do ~ 1, data = HCMC, family = "binomial")

# full models for forward-backward selection
dep_class_full <- glm(full_dep_class_form, data = HCMC, family = "binomial")
sub_class_full <- glm(full_sub_class_form, data = HCMC, family = "binomial")
tth_class_full <- glm(full_tth_class_form, data = HCMC, family = "binomial")
som_class_full <- glm(full_som_class_form, data = HCMC, family = "binomial")

dep_class_hsw <- stepAIC(object = dep_class_int,
                      scope = list(upper = dep_class_full, lower = dep_class_int),
                      direction = "both", trace = FALSE)
sub_class_hsw <- stepAIC(object = sub_class_int,
                      scope = list(upper = sub_class_full, lower = sub_class_int),
                      direction = "both", trace = FALSE)
tth_class_hsw <- stepAIC(object = tth_class_int,
                      scope = list(upper = tth_class_full, lower = tth_class_int),
                      direction = "both", trace = FALSE)
som_class_hsw <- stepAIC(object = som_class_int,
                      scope = list(upper = som_class_full, lower = som_class_int),
                      direction = "both", trace = FALSE)

# Test models on external data and consolidate results -----
coef_names <- c("(Intercept)", "rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2",
"rfRC3", "rfRC4",
               "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
"rfCOG",
               "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
"rfMSF",

```

```

      "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
"rfDSF",
      "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")

fsw_acc_mdls <- list(dep_class_fsw, sub_class_fsw, tht_class_fsw, som_class_fsw)
bsw_acc_mdls <- list(dep_class_bsw, sub_class_bsw, tht_class_bsw, som_class_bsw)
hsw_acc_mdls <- list(dep_class_hsw, sub_class_hsw, tht_class_hsw, som_class_hsw)
fsw_roc_mdls <- list(dep_roc_fsw, sub_roc_fsw, tht_roc_fsw, som_roc_fsw)
bsw_roc_mdls <- list(dep_roc_bsw, sub_roc_bsw, tht_roc_bsw, som_roc_bsw)

fsw_class_out <- matrix(NA,
                        nrow = length(coef_nams) + 8,
                        ncol = length(fsw_acc_mdls))
rownames(fsw_class_out) <- c(coef_nams,
                             "Acc-Train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                             "AUC-Train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(fsw_class_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

bsw_class_out <- hsw_class_out <- fsw_class_out

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

# Consolidate forward stepwise logistic regression results
for(i in 1:length(fsw_acc_mdls)){
  curr_mdl <- fsw_acc_mdls[[i]]
  curr_roc <- fsw_roc_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  coef_idx <- which(coef_nams %in% names(coefs))
  n_coefs <- length(coef_nams)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_cv <- curr_mdl$results$Accuracy
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_roc$results$ROC
  AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
  AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

  # Place coefficients in correct cells
  for(j in 1:length(coefs)){
    par_nams <- names(coefs)
    coef_idx <- which(rownames(fsw_class_out) == par_nams[j])
    fsw_class_out[coef_idx, i] <- coefs[j]
  }

  fsw_class_out[n_coefs + 1, i] <- acc_train
  fsw_class_out[n_coefs + 2, i] <- acc_cv
  fsw_class_out[n_coefs + 3, i] <- acc_ev_VAMC
  fsw_class_out[n_coefs + 4, i] <- acc_ev_PP
  fsw_class_out[n_coefs + 5, i] <- AUC_train
  fsw_class_out[n_coefs + 6, i] <- AUC_cv
  fsw_class_out[n_coefs + 7, i] <- AUC_ev_VAMC

```

```

    fsw_class_out[n_coefs + 8, i] <- AUC_ev_PP
  }

# Consolidate backward stepwise logistic regression results
for(i in 1:length(bsw_acc_mdls)){
  curr_mdl <- bsw_acc_mdls[[i]]
  curr_roc <- bsw_roc_mdls[[i]]
  coefs <- curr_mdl$finalModel$coefficients
  coef_idx <- which(coef_nams %in% names(coefs))
  n_coefs <- length(coef_nams)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_cv <- curr_mdl$results$Accuracy
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_roc$results$ROC
  AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
  AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

  # Place coefficients in correct cells
  for(j in 1:length(coefs)){
    par_nams <- names(coefs)
    coef_idx <- which(rownames(bsw_class_out) == par_nams[j])
    bsw_class_out[coef_idx, i] <- coefs[j]
  }

  bsw_class_out[n_coefs + 1, i] <- acc_train
  bsw_class_out[n_coefs + 2, i] <- acc_cv
  bsw_class_out[n_coefs + 3, i] <- acc_ev_VAMC
  bsw_class_out[n_coefs + 4, i] <- acc_ev_PP
  bsw_class_out[n_coefs + 5, i] <- AUC_train
  bsw_class_out[n_coefs + 6, i] <- AUC_cv
  bsw_class_out[n_coefs + 7, i] <- AUC_ev_VAMC
  bsw_class_out[n_coefs + 8, i] <- AUC_ev_PP
}

# Consolidate hybrid stepwise logistic regression results
hsw_class_ints <- list(dep_class_int, sub_class_int, tht_class_int, som_class_int)
hsw_class_fulls <- list(dep_class_full, sub_class_full, tht_class_full,
  som_class_full)
for(i in 1:length(hsw_acc_mdls)){
  curr_mdl <- hsw_acc_mdls[[i]]
  curr_int <- hsw_class_ints[[i]]
  curr_full <- hsw_class_fulls[[i]]
  coefs <- curr_mdl$coefficients
  n_coefs <- length(coef_nams)

  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  train_outcome <- curr_mdl$model[1] %>% unlist %>% as.factor

```

```

train_pred_probs <- predict(curr_mdl, HCMC, type = "response")
train_pred_class <- rep(0, nrow(HCMC))
train_pred_class[train_pred_probs >= .5] <- 1
train_pred_class <- as.factor(train_pred_class)
levels(train_pred_class) <- levels(train_outcome)

VA_pred_probs <- predict(curr_mdl, VAMC, type = "response")
VA_pred_class <- rep(0, nrow(VAMC))
VA_pred_class[VA_pred_probs >= .5] <- 1
VA_pred_class <- as.factor(VA_pred_class)
levels(VA_pred_class) <- levels(train_outcome)

PP_pred_probs <- predict(curr_mdl, ppath, type = "response")
PP_pred_class <- rep(0, nrow(ppath))
PP_pred_class[PP_pred_probs >= .5] <- 1
PP_pred_class <- as.factor(PP_pred_class)
levels(PP_pred_class) <- levels(train_outcome)

acc_train <- confusionMatrix(train_pred_class, train_outcome)$overall['Accuracy']
acc_ev_VAMC <- confusionMatrix(VA_pred_class, test_outcome_VAMC)$overall['Accuracy']
acc_ev_PP <- confusionMatrix(PP_pred_class, test_outcome_PP)$overall['Accuracy']
AUC_train <- roc(train_outcome, train_pred_probs)$auc
AUC_ev_VAMC <- roc(test_outcome_VAMC, VA_pred_probs)$auc
AUC_ev_PP <- roc(test_outcome_PP, PP_pred_probs)$auc

# Get cross-validated performance
cv_rslts <- step_cv_class(curr_int, curr_full, data = HCMC)
acc_cv <- cv_rslts$Acc
AUC_cv <- cv_rslts$AUC

# Place coefficients in correct cells
for(j in 1:length(coefs)){
  par_nams <- names(coefs)
  coef_idx <- which(rownames(hsw_class_out) == par_nams[j])
  hsw_class_out[coef_idx, j] <- coefs[j]
}

hsw_class_out[n_coefs + 1, 1] <- acc_train
hsw_class_out[n_coefs + 2, 1] <- acc_cv
hsw_class_out[n_coefs + 3, 1] <- acc_ev_VAMC
hsw_class_out[n_coefs + 4, 1] <- acc_ev_PP
hsw_class_out[n_coefs + 5, 1] <- AUC_train
hsw_class_out[n_coefs + 6, 1] <- AUC_cv
hsw_class_out[n_coefs + 7, 1] <- AUC_ev_VAMC
hsw_class_out[n_coefs + 8, 1] <- AUC_ev_PP
}

# Export stepwise selection logistic regression results
setwd(outwd)
write.csv(fsw_class_out, "Forward Stepwise Classification Output.csv")
write.csv(bsw_class_out, "Backward Stepwise Classification Output.csv")
write.csv(hsw_class_out, "Hybrid Stepwise Classification Output.csv")
setwd(wd)

# Lasso Linear Regression using glmnet
#=====

# Train models using full training data
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
           "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
           "rfCOG",

```

```

      "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
"rfMSF",
      "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
"rfDSF",
      "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

set.seed(23001)
full_dep_lasso_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct1, alpha = 1)
set.seed(23002)
full_sub_lasso_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct4, alpha = 1)
set.seed(23003)
full_tht_lasso_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct5, alpha = 1)
set.seed(23004)
full_pai_lasso_reg <- cv.glmnet(x = train_mat, y = HCMC$msefct8, alpha = 1)

# Export lasso regression results
full_lasso_mdls <- list(full_dep_lasso_reg, full_sub_lasso_reg, full_tht_lasso_reg,
full_pai_lasso_reg)

full_lasso_out <- matrix(NA,
                        nrow = nrow(full_lasso_mdls[[1]]$glmnet.fit$beta) + 7,
                        ncol = length(full_lasso_mdls))
rownames(full_lasso_out) <- c("Intercept",
rownames(full_lasso_mdls[[1]]$glmnet.fit$beta),
                        "R2-train", "R2-CV", "R2-EV",
                        "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(full_lasso_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

test_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()

for(i in 1:length(full_lasso_mdls)){
  curr_mdl <- full_lasso_mdls[[i]]
  lambda_idx <- which(curr_mdl$glmnet.fit$lambda == curr_mdl$lambda.min)
  coefs <- curr_mdl$glmnet.fit$beta[, lambda_idx]
  n_coefs <- length(coefs)

  train_outcome <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat)
  test_pred <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_mat)
  test_outcome <- VAMC[, reg_outcomes[i]]

  R2_train <- R2(train_outcome, HCMC[, reg_outcomes[i]])
  R2_cv <- glmnet_cv_R2(curr_mdl, x = train_mat, y = as.matrix(HCMC[,
reg_outcomes[i]]))
  R2_ev <- R2(test_pred, VAMC[, reg_outcomes[i]])
  rmse_train <- RMSE(train_outcome, HCMC[, reg_outcomes[i]])
  rmse_cv <- glmnet_cv_rmse(curr_mdl)
  rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

  full_lasso_out[1, i] <- curr_mdl$glmnet.fit$a0[lambda_idx]
  full_lasso_out[2:(n_coefs + 1), i] <- coefs
  full_lasso_out[n_coefs + 2, i] <- R2_train
  full_lasso_out[n_coefs + 3, i] <- R2_cv
  full_lasso_out[n_coefs + 4, i] <- R2_ev
  full_lasso_out[n_coefs + 5, i] <- rmse_train
  full_lasso_out[n_coefs + 6, i] <- rmse_cv
  full_lasso_out[n_coefs + 7, i] <- rmse_ev
}

```

```

setwd(outwd)
write.csv(full_lasso_out, "Lasso Linear Regression Output.csv")
setwd(wd)

# Lasso Logistic Regression using glmnet
#=====

# Train models using full training data
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
           "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
           "rfCOG",
           "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
           "rfMSF",
           "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
           "rfDSF",
           "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()

set.seed(24001)
full_dep_llr_class <- cv.glmnet(x = train_mat, y = HCMC$dep_do, alpha = 1,
                               family = "binomial")

set.seed(24002)
full_sub_llr_class <- cv.glmnet(x = train_mat, y = HCMC$sub_do, alpha = 1,
                               family = "binomial")

set.seed(24003)
full_tht_llr_class <- cv.glmnet(x = train_mat, y = HCMC$thought_do, alpha = 1,
                               family = "binomial")

set.seed(24004)
full_som_llr_class <- cv.glmnet(x = train_mat, y = HCMC$som_do, alpha = 1,
                               family = "binomial")

# Test ridge logistic models on external data and consolidate results
full_llr_mdls <- list(full_dep_llr_class, full_sub_llr_class, full_tht_llr_class,
                     full_som_llr_class)

full_llr_out <- matrix(NA,
                      nrow = nrow(full_llr_mdls[[1]]$glmnet.fit$beta) + 9,
                      ncol = length(full_llr_mdls))
rownames(full_llr_out) <- c("Intercept", rownames(full_llr_mdls[[1]]$glmnet.fit$beta),
                          "Acc-train", "Acc-CV", "Acc-EV-VAMC", "Acc-EV-PP",
                          "AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(full_llr_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

test_VAMC_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()
test_PP_mat <- dplyr::select(ppath, one_of(preds)) %>%
  as.matrix()

for(i in 1:length(full_llr_mdls)){
  curr_mdl <- full_llr_mdls[[i]]
  lambda_idx <- which(curr_mdl$glmnet.fit$lambda == curr_mdl$lambda.min)
  coefs <- curr_mdl$glmnet.fit$beta[, lambda_idx]
  n_coefs <- length(coefs)

  train_pred_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat,
                             type = "response")
  train_pred_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = train_mat,
                             type = "class") %>% as.factor()

```

```

test_pred_VAMC_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx =
test_VAMC_mat,
                                type = "response")
test_pred_VAMC_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx =
test_VAMC_mat,
                                type = "class") %>% as.factor()
test_pred_PP_probs <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_PP_mat,
                                type = "response")
test_pred_PP_class <- predict(curr_mdl, s = curr_mdl$lambda.min, newx = test_PP_mat,
                                type = "class") %>% as.factor()

train_obs <- HCMC[, class_outcomes[i]] %>% as.data.frame()
train_obs <- as.vector(train_obs[, 1]) %>% as.factor()
test_obs_VAMC <- VAMC[, class_outcomes[i]] %>% as.data.frame()
test_obs_VAMC <- as.vector(test_obs_VAMC[,1]) %>% as.factor()
test_obs_PP <- ppath[, class_outcomes[i]] %>% as.data.frame()
test_obs_PP <- as.vector(test_obs_PP[,1]) %>% as.factor()

seed <- 9000 + i
set.seed(seed)
cv_rslts <- glmnet_cv_class_stats(curr_mdl, x = train_mat, y = HCMC[,
class_outcomes[i]])

acc_train <- confusionMatrix(train_pred_class, train_obs)$overall['Accuracy']
acc_cv <- cv_rslts$Acc
acc_ev_VAMC <- confusionMatrix(test_pred_VAMC_class,
test_obs_VAMC)$overall['Accuracy']
acc_ev_PP <- confusionMatrix(test_pred_PP_class, test_obs_PP)$overall['Accuracy']
AUC_train <- roc(train_obs, train_pred_probs)$auc
AUC_cv <- cv_rslts$AUC
AUC_ev_VAMC <- roc(test_obs_VAMC, test_pred_VAMC_probs)$auc
AUC_ev_PP <- roc(test_obs_PP, test_pred_PP_probs)$auc

full_llr_out[1, i] <- curr_mdl$glmnet.fit$a0[lambda_idx]
full_llr_out[2:(n_coefs + 1), i] <- coefs
full_llr_out[n_coefs + 2, i] <- acc_train
full_llr_out[n_coefs + 3, i] <- acc_cv
full_llr_out[n_coefs + 4, i] <- acc_ev_VAMC
full_llr_out[n_coefs + 5, i] <- acc_ev_PP
full_llr_out[n_coefs + 6, i] <- AUC_train
full_llr_out[n_coefs + 7, i] <- AUC_cv
full_llr_out[n_coefs + 8, i] <- AUC_ev_VAMC
full_llr_out[n_coefs + 9, i] <- AUC_ev_PP
}

setwd(outwd)
write.csv(full_llr_out, "Lasso Logistic Regression Output.csv")
setwd(wd)

# Full-Scale Random Forests Regression
#=====

# Train models using full training data -----

# Define random forests training control. This will allow the algorithm to explore a
# greater range of tuning parameters.
rf_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random")

set.seed(25001)
full_dep_rf_train <- train(form = full_dep_reg_form, data = HCMC,

```

```

        trControl = rf_control, method = "rf",
        tuneLength = 15)
set.seed(25002)
full_sub_rf_train <- train(form = full_sub_reg_form, data = HCMC,
        trControl = rf_control, method = "rf",
        tuneLength = 15)
set.seed(25003)
full_tht_rf_train <- train(form = full_tht_reg_form, data = HCMC,
        trControl = rf_control, method = "rf",
        tuneLength = 15)
set.seed(25004)
full_pai_rf_train <- train(form = full_pai_reg_form, data = HCMC,
        trControl = rf_control, method = "rf",
        tuneLength = 15)

# Export random forests regression results
full_rf_mdls <- list(full_dep_rf_train, full_sub_rf_train, full_tht_rf_train,
full_pai_rf_train)

full_rf_out <- matrix(NA,
        nrow = length(full_rf_mdls[[1]]$finalModel$importance) + 3,
        ncol = length(full_rf_mdls))
rownames(full_rf_out) <- c(rownames(full_rf_mdls[[1]]$finalModel$importance),
        "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(full_rf_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(full_rf_mdls)){
    curr_mdl <- full_rf_mdls[[i]]
    importance <- curr_mdl$finalModel$importance
    n_coefs <- length(importance)

    best_mtry <- curr_mdl$bestTune %>% as.numeric()
    best_idx <- which(curr_mdl$results$mtry == best_mtry)
    best_mtry_RMSE <- curr_mdl$results$RMSE[best_idx]

    train_outcome <- curr_mdl$trainingData$outcome
    test_pred <- predict.train(curr_mdl, VAMC)
    test_outcome <- VAMC[, reg_outcomes[i]]

    rmse_train <- RMSE(predict(curr_mdl, HCMC), HCMC[, reg_outcomes[i]])
    rmse_cv <- best_mtry_RMSE
    rmse_ev <- RMSE(test_pred, VAMC[, reg_outcomes[i]])

    full_rf_out[1:n_coefs, i] <- importance
    full_rf_out[n_coefs + 1, i] <- rmse_train
    full_rf_out[n_coefs + 2, i] <- rmse_cv
    full_rf_out[n_coefs + 3, i] <- rmse_ev
}

# Export random forests regression results
setwd(outwd)
write.csv(full_rf_out, "Full-Scale Random Forests Regression Output.csv")
setwd(wd)

# Full-Scale Random Forests Classification
#=====

# Define random forests training control. This will allow the algorithm to explore a

```



```

# greater range of tuning parameters and force it to use ROC as its optimizing
function.
rfc_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random",
                           classProbs = TRUE, summaryFunction = twoClassSummary)

# Train RF classification models. Use AUC as performance criterion (preferred over
accuracy
# in this case due to class imbalances)
set.seed(260001)
full_dep_rfc_train <- train(form = full_dep_class_form, data = HCMC,
                           trControl = rfc_control, method = "rf",
                           tuneLength = 15, metric = "ROC")

set.seed(260002)
full_sub_rfc_train <- train(form = full_sub_class_form, data = HCMC,
                           trControl = rfc_control, method = "rf",
                           tuneLength = 15, metric = "ROC")

set.seed(260003)
full_tht_rfc_train <- train(form = full_tht_class_form, data = HCMC,
                           trControl = rfc_control, method = "rf",
                           tuneLength = 15, metric = "ROC")

set.seed(260004)
full_som_rfc_train <- train(form = full_som_class_form, data = HCMC,
                           trControl = rfc_control, method = "rf",
                           tuneLength = 15, metric = "ROC")

# Export random forests regression results [UPDATE]
full_rfc_mdls <- list(full_dep_rfc_train, full_sub_rfc_train, full_tht_rfc_train,
full_som_rfc_train)

full_rfc_out <- matrix(NA,
                      nrow = length(full_rfc_mdls[[1]]$finalModel$importance) + 7,
                      ncol = length(full_rfc_mdls))
rownames(full_rfc_out) <- c(rownames(full_rfc_mdls[[1]]$finalModel$importance),
"Acc-train", "Acc-EV-VAMC", "Acc-EV-PP",
"AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(full_rfc_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(full_rfc_mdls)){
  curr_mdl <- full_rfc_mdls[[i]]
  importance <- curr_mdl$finalModel$importance
  n_coefs <- length(importance)

  best_mtry <- curr_mdl$bestTune %>% as.numeric()
  best_idx <- which(curr_mdl$results$mtry == best_mtry)

  train_outcome <- curr_mdl$trainingData$outcome
  train_pred <- predict.train(curr_mdl, HCMC)
  train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
  test_VAMC <- predict.train(curr_mdl, VAMC)
  test_PP <- predict.train(curr_mdl, ppath)
  test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
  test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
  test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
  test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

  acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
  acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
  acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
  AUC_train <- roc(train_outcome, train_probs$Pos)$auc
  AUC_cv <- curr_mdl$results$ROC[best_idx]

```

```

AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

full_rfc_out[1:n_coefs, i] <- importance
full_rfc_out[n_coefs + 1, i] <- acc_train
full_rfc_out[n_coefs + 2, i] <- acc_ev_VAMC
full_rfc_out[n_coefs + 3, i] <- acc_ev_PP
full_rfc_out[n_coefs + 4, i] <- AUC_train
full_rfc_out[n_coefs + 5, i] <- AUC_cv
full_rfc_out[n_coefs + 6, i] <- AUC_ev_VAMC
full_rfc_out[n_coefs + 7, i] <- AUC_ev_PP
}

# Export random forests regression results
setwd(outwd)
write.csv(full_rfc_out, "Full-Scale Random Forests Classification Output.csv")
setwd(wd)

# Full-scale support vector regression
#=====

# Train models using full set of training data
set.seed(27001)
full_dep_svr_train <- train(form = full_dep_reg_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = cv_control, method = "svmRadial",
                           tuneLength = 14)

set.seed(27002)
full_sub_svr_train <- train(form = full_sub_reg_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = cv_control, method = "svmRadial",
                           tuneLength = 14)

set.seed(27003)
full_tht_svr_train <- train(form = full_tht_reg_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = cv_control, method = "svmRadial",
                           tuneLength = 14)

set.seed(27004)
full_pai_svr_train <- train(form = full_pai_reg_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = cv_control, method = "svmRadial",
                           tuneLength = 14)

# Evaluate SVR models on external data and consolidate results

full_svr_mdls <- list(full_dep_svr_train, full_sub_svr_train, full_tht_svr_train,
full_pai_svr_train)

full_svr_out <- matrix(NA,
                      nrow = 3,
                      ncol = length(full_svr_mdls))
rownames(full_svr_out) <- c("RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(full_svr_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(full_svr_mdls)){
  curr_md1 <- full_svr_mdls[[i]]

  best_C <- curr_md1$bestTune$C
  best_idx <- which(curr_md1$results$C == best_C)
  best_C_RMSE <- curr_md1$results$RMSE[best_idx]

```

```

train_outcome <- curr_mdl$trainingData$outcome
test_pred <- predict.train(curr_mdl, VAMC)
test_outcome <- VAMC[, reg_outcomes[i]]

rmse_train <- RMSE(predict(curr_mdl, HCMC), train_outcome)
rmse_cv <- best_C_RMSE
rmse_ev <- RMSE(test_pred, test_outcome)

full_svr_out[1, i] <- rmse_train
full_svr_out[2, i] <- rmse_cv
full_svr_out[3, i] <- rmse_ev
}

# Export SVR results
setwd(outwd)
write.csv(full_svr_out, "Full-Scale Support Vector Regression Output.csv")
setwd(wd)

# Full-Scale Support Vector Machines
#=====

# Train models using full set of training data
set.seed(28001)
full_dep_svm_train <- train(form = full_dep_class_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = roc_control, method = "svmRadial",
                           tuneLength = 14)

set.seed(28002)
full_sub_svm_train <- train(form = full_sub_class_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = roc_control, method = "svmRadial",
                           tuneLength = 14)

set.seed(28003)
full_tht_svm_train <- train(form = full_tht_class_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = roc_control, method = "svmRadial",
                           tuneLength = 14)

set.seed(28004)
full_som_svm_train <- train(form = full_som_class_form, data = HCMC,
                           preProc = c("center", "scale"),
                           trControl = roc_control, method = "svmRadial",
                           tuneLength = 14)

# Evaluate SVM models on external data and consolidate results

full_svm_mdls <- list(full_dep_svm_train, full_sub_svm_train, full_tht_svm_train,
full_som_svm_train)

full_svm_out <- matrix(NA,
                      nrow = 7,
                      ncol = length(full_svm_mdls))
rownames(full_svm_out) <- c("Acc-train", "Acc-EV-VAMC", "Acc-EV-PP",
                          "AUC-train", "AUC-CV", "AUC-EV-VAMC", "AUC-EV-PP")
colnames(full_svm_out) <- c("Depression", "Sub Abuse", "Thought", "Somatic")

class_outcomes <- c("dep_do", "sub_do", "thought_do", "som_do")

for(i in 1:length(full_svm_mdls)){
  curr_mdl <- full_svm_mdls[[i]]

  best_C <- curr_mdl$bestTune$C

```

```

best_idx <- which(curr_mdl$results$C == best_C)

train_outcome <- curr_mdl$trainingData$outcome
test_pred <- predict.train(curr_mdl, VAMC)
test_outcome <- VAMC[, reg_outcomes[i]]
train_pred <- predict.train(curr_mdl, HCMC)
train_probs <- predict.train(curr_mdl, HCMC, type = "prob")
test_VAMC <- predict.train(curr_mdl, VAMC)
test_PP <- predict.train(curr_mdl, ppath)
test_probs_VAMC <- predict.train(curr_mdl, VAMC, type = "prob")
test_probs_PP <- predict.train(curr_mdl, ppath, type = "prob")
test_outcome_VAMC <- VAMC[, class_outcomes[i]] %>% as.matrix
test_outcome_PP <- ppath[, class_outcomes[i]] %>% as.matrix

acc_train <- confusionMatrix(train_pred, train_outcome)$overall['Accuracy']
acc_ev_VAMC <- confusionMatrix(test_VAMC, test_outcome_VAMC)$overall['Accuracy']
acc_ev_PP <- confusionMatrix(test_PP, test_outcome_PP)$overall['Accuracy']
AUC_train <- roc(train_outcome, train_probs$Pos)$auc
AUC_cv <- curr_mdl$results$ROC[best_idx]
AUC_ev_VAMC <- roc(test_outcome_VAMC, test_probs_VAMC$Pos)$auc
AUC_ev_PP <- roc(test_outcome_PP, test_probs_PP$Pos)$auc

full_svm_out[1, i] <- acc_train
full_svm_out[2, i] <- acc_ev_VAMC
full_svm_out[3, i] <- acc_ev_PP
full_svm_out[4, i] <- AUC_train
full_svm_out[5, i] <- AUC_cv
full_svm_out[6, i] <- AUC_ev_VAMC
full_svm_out[7, i] <- AUC_ev_PP
}

# Export SVM results
setwd(outwd)
write.csv(full_svm_out, "Full-Scale Support Vector Machine Output.csv")
setwd(wd)

#####
# Descriptives and Secondary Analyses      #
#####

# Check distributions of criterion variables
# -----

# Plot distributions of count criteria

# Get plots using combined inpatient sets
inpt <- valid_only(MN_inpt_full)

plot_dep <- ggplot(inpt, aes(x = msefct1)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Depression Score", y = "Frequency")
plot_sub <- ggplot(inpt, aes(x = msefct4)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Substance Abuse Score", y = "Frequency")
plot_del <- ggplot(inpt, aes(x = msefct5)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Delusions Score", y = "Frequency")
plot_pain <- ggplot(inpt, aes(x = msefct8)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Pain Score", y = "Frequency")

# Get plots for HCMC

```

```

plot_dep_HCMC <- ggplot(HCMC, aes(x = msefct1)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Depression Score", y = "Frequency")
plot_sub_HCMC <- ggplot(HCMC, aes(x = msefct4)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Sub. Abuse Score", y = "Frequency")
plot_del_HCMC <- ggplot(HCMC, aes(x = msefct5)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Delusions Score", y = "Frequency")
plot_pain_HCMC <- ggplot(HCMC, aes(x = msefct8)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Pain Score", y = "Frequency")

# Get plots for VAMC
plot_dep_VAMC <- ggplot(VAMC, aes(x = msefct1)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Depression Score", y = "Frequency")
plot_sub_VAMC <- ggplot(VAMC, aes(x = msefct4)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Sub. Abuse Score", y = "Frequency")
plot_del_VAMC <- ggplot(VAMC, aes(x = msefct5)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Delusions Score", y = "Frequency")
plot_pain_VAMC <- ggplot(VAMC, aes(x = msefct8)) +
  geom_histogram(binwidth = 1, position = "identity") +
  labs(x = "Pain Score", y = "Frequency")

# Output distribution histograms for manuscript
ppi <- 300
setwd(outwd)
png("MSE Scales - Combined Inpatient.png", width = 6 * ppi, height = 4 * ppi, res =
ppi)
grid.arrange(plot_dep, plot_sub, plot_del, plot_pain,
             ncol = 2, nrow = 2)
dev.off()
png("MSE Scales - HCMC.png", width = 6 * ppi, height = 4 * ppi, res = ppi)
grid.arrange(plot_dep_HCMC, plot_sub_HCMC, plot_del_HCMC, plot_pain_HCMC,
             ncol = 2, nrow = 2)
dev.off()
png("MSE Scales - VAMC.png", width = 6 * ppi, height = 4 * ppi, res = ppi)
grid.arrange(plot_dep_VAMC, plot_sub_VAMC, plot_del, plot_pain_VAMC,
             ncol = 2, nrow = 2)
dev.off()
setwd(wd)

# Output distribution histograms for Symposium presentation.
ppi <- 300
setwd(outwd)
png("MSE Scales - Presentation.png", width = 6 * ppi, height = 4 * ppi, res = ppi)
grid.arrange(plot_dep_HCMC, plot_sub_HCMC, plot_del_HCMC,
             plot_dep_VAMC, plot_sub_VAMC, plot_del_VAMC,
             ncol = 3, nrow = 2)
dev.off()
setwd(wd)

# Check diagnostic group base rates -----
HCMC_dep_br <- calc_br(HCMC, "dep_do_num") # .46
HCMC_sub_br <- calc_br(HCMC, "sub_do_num") # .44
HCMC_tht_br <- calc_br(HCMC, "thought_do_num") # .17
HCMC_som_br <- calc_br(HCMC, "som_do_num") # .01

VAMC_dep_br <- calc_br(VAMC, "dep_do_num") # .39

```

```

VAMC_sub_br <- calc_br(VAMC, "sub_do_num") # .45
VAMC_tht_br <- calc_br(VAMC, "thought_do_num") # .14
VAMC_som_br <- calc_br(VAMC, "som_do_num") # .02

ppath_dep_br <- calc_br(ppath, "dep_do_num") # .24
ppath_sub_br <- calc_br(ppath, "sub_do_num") # .14
ppath_tht_br <- calc_br(ppath, "thought_do_num") # .02
ppath_som_br <- calc_br(ppath, "som_do_num") # <.01

# Plot base rates -----
HCMC_brs <- c(HCMC_dep_br, HCMC_sub_br, HCMC_tht_br, HCMC_som_br)
VAMC_brs <- c(VAMC_dep_br, VAMC_sub_br, VAMC_tht_br, VAMC_som_br)
ppath_brs <- c(ppath_dep_br, ppath_sub_br, ppath_tht_br, ppath_som_br)
disorders <- c("Depressive Disorder", "Substance Use Disorder",
               "Thought Disorder", "Somatic Disorder")
disorders <- factor(disorders, levels = disorders)

rates <- data.frame(disorders, HCMC_brs, VAMC_brs, ppath_brs)
colnames(rates) <- c("Diagnosis", "County Inpt.", "VA Inpt.", "Community Outpt.")

# Convert to long data format for visualization
rates_long <- gather(rates, Site, Rate, "County Inpt.":"Community Outpt.", -Diagnosis,
                    factor_key = TRUE)
rates_conf_long <- gather(rates_conf, Site, Rate, "County Inpt.":"VA Inpt.", -
Diagnosis,
                        factor_key = TRUE)

# Black and white bar graph of base rates for manuscript
br_plot <- ggplot(rates_long, aes(x = Site, y = Rate, fill = Diagnosis)) +
  geom_bar(stat = "identity", position = "dodge") +
  theme_bw() +
  scale_fill_grey()

# Output base rate bar plots
ppi <- 300
setwd(outwd)
png("Base Rates - Manuscript.png", width = 6 * ppi, height = 4 * ppi, res = ppi)
br_plot
dev.off()
dev.off()
setwd(wd)

# Correlational analysis for regression variables
#
=====

# Prepare datasets for correlational analysis
predictors <- c("rfEID", "rfTHD", "rfBXD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
               "rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC",
               "rfNUC", "rfCOG",
               "rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP",
               "rfBRF", "rfMSF",
               "rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV",
               "rfSHY", "rfDSF",
               "rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")
nice_RF_names <- c("EID", "THD", "BXD", "RCd", "RC1", "RC2", "RC3", "RC4",
                  "RC6", "RC7", "RC8", "RC9", "MLS", "GIC", "HPC", "NUC", "COG",
                  "SUI", "HLP", "SFD", "NFC", "STW", "AXY", "ANP", "BRF", "MSF",
                  "JCP", "SUB", "AGG", "ACT", "FML", "IPP", "SAV", "SHY", "DSF",

```

```

      "AES", "MEC", "AGGR", "PSYC", "DISC", "NEGE", "INTR")
nice_reg_names <- c("MSE-Depressed", "MSE-SubAbuse", "MSE-Delusions", "MSE-Pain")

HCMC_RF_scores <- dplyr::select(HCMC, one_of(predictors))
VAMC_RF_scores <- dplyr::select(VAMC, one_of(predictors))
ppath_RF_scores <- dplyr::select(ppath, one_of(predictors))
HCMC_reg_out <- dplyr::select(HCMC, one_of(reg_outcomes))
VAMC_reg_out <- dplyr::select(VAMC, one_of(reg_outcomes))
colnames(HCMC_RF_scores) <- colnames(VAMC_RF_scores) <- colnames(ppath_RF_scores) <-
nice_RF_names
colnames(HCMC_reg_out) <- colnames(VAMC_reg_out) <- nice_reg_names

HCMC_corr_dat <- dplyr::select(HCMC, one_of(c(predictors, reg_outcomes)))
VAMC_corr_dat <- dplyr::select(VAMC, one_of(c(predictors, reg_outcomes)))
colnames(HCMC_corr_dat) <- colnames(VAMC_corr_dat) <- c(nice_RF_names, nice_reg_names)

# Network plots
HCMC_net <- HCMC_corr_dat %>% correlate() %>% network_plot(min_cor = .4)
VAMC_net <- VAMC_corr_dat %>% correlate() %>% network_plot(min_cor = .4)
ppi <- 300
setwd(outwd)
png("Network - HCMC.png", width = 6 * ppi, height = 4 * ppi, res = ppi)
HCMC_net
dev.off()
png("Network - VAMC.png", width = 6 * ppi, height = 4 * ppi, res = ppi)
VAMC_net
dev.off()
setwd(wd)

# MMPI-2-RF intercorrelations
HCMC_RF_corrs <- correlate(HCMC_RF_scores)
VAMC_RF_corrs <- correlate(VAMC_RF_scores)
ppath_RF_corrs <- correlate(ppath_RF_scores)

# RRF intercorrelations
HCMC_RRF_corrs <- correlate(HCMC_reg_out)
VAMC_RRF_corrs <- correlate(VAMC_reg_out)

# Correlations between MMPI-2-RF scales and RRF scales
HCMC_corrs <- cor(HCMC_RF_scores, HCMC_reg_out)
VAMC_corrs <- cor(VAMC_RF_scores, VAMC_reg_out)

# Prepare correlations for output
HCMC_RF_corrs_pretty <- HCMC_RF_corrs %>% fashion()
VAMC_RF_corrs_pretty <- VAMC_RF_corrs %>% fashion()
ppath_RF_corrs_pretty <- ppath_RF_corrs %>% fashion()
HCMC_RRF_corrs_pretty <- HCMC_RRF_corrs %>% fashion()
VAMC_RRF_corrs_pretty <- VAMC_RRF_corrs %>% fashion()
HCMC_corrs_pretty <- HCMC_corrs %>% fashion()
VAMC_corrs_pretty <- VAMC_corrs %>% fashion()
rownames(HCMC_corrs_pretty) <- rownames(VAMC_corrs_pretty) <- predictors

# Output correlations
setwd(outwd)
write.csv(HCMC_RF_corrs_pretty, "MMPI-2-RF Intercorrelations - HCMC.csv")
write.csv(VAMC_RF_corrs_pretty, "MMPI-2-RF Intercorrelations - VAMC.csv")
write.csv(ppath_RF_corrs_pretty, "MMPI-2-RF Intercorrelations - Portage Path.csv")
write.csv(HCMC_RRF_corrs_pretty, "RRF Scale Intercorrelations - HCMC.csv")
write.csv(VAMC_RRF_corrs_pretty, "RRF Scale Intercorrelations - VAMC.csv")
write.csv(HCMC_corrs_pretty, "Correlations Between RF and RRF - HCMC")
write.csv(VAMC_corrs_pretty, "Correlations Between RF and RRF - VAMC")
setwd(wd)

```

```

# ROC analysis for classification variables
#
=====

HCMC_dep_auc <- rf_auc(HCMC, "dep_do")
HCMC_sub_auc <- rf_auc(HCMC, "sub_do")
HCMC_tht_auc <- rf_auc(HCMC, "thought_do")
HCMC_som_auc <- rf_auc(HCMC, "som_do")

VAMC_dep_auc <- rf_auc(VAMC, "dep_do")
VAMC_sub_auc <- rf_auc(VAMC, "sub_do")
VAMC_tht_auc <- rf_auc(VAMC, "thought_do")
VAMC_som_auc <- rf_auc(VAMC, "som_do")

ppath_dep_auc <- rf_auc(ppath, "dep_do")
ppath_sub_auc <- rf_auc(ppath, "sub_do")
ppath_tht_auc <- rf_auc(ppath, "thought_do")
ppath_som_auc <- rf_auc(ppath, "som_do")

# Compile AUC stats by site
HCMC_AUCs <- matrix(NA, nrow = nrow(HCMC_dep_auc), ncol = 4)
rownames(HCMC_AUCs) <- HCMC_dep_auc$Scale
colnames(HCMC_AUCs) <- c("Depressive Dx", "Substance Use Dx", "Thought Dx", "Somatic
Dx")
VAMC_AUCs <- ppath_AUCs <- HCMC_AUCs

HCMC_AUCs[, 1] <- HCMC_dep_auc$AUC
HCMC_AUCs[, 2] <- HCMC_sub_auc$AUC
HCMC_AUCs[, 3] <- HCMC_tht_auc$AUC
HCMC_AUCs[, 4] <- HCMC_som_auc$AUC

VAMC_AUCs[, 1] <- VAMC_dep_auc$AUC
VAMC_AUCs[, 2] <- VAMC_sub_auc$AUC
VAMC_AUCs[, 3] <- VAMC_tht_auc$AUC
VAMC_AUCs[, 4] <- VAMC_som_auc$AUC

ppath_AUCs[, 1] <- ppath_dep_auc$AUC
ppath_AUCs[, 2] <- ppath_sub_auc$AUC
ppath_AUCs[, 3] <- ppath_tht_auc$AUC
ppath_AUCs[, 4] <- ppath_som_auc$AUC

# Output AUC statistics
setwd(outwd)
write.csv(HCMC_AUCs, "AUCs - HCMC.csv")
write.csv(VAMC_AUCs, "AUCs - VAMC.csv")
write.csv(ppath_AUCs, "AUCs - Portage Path.csv")
setwd(wd)

# Plot observed vs. predicted values for regression
#-----
-----

# Extract observed RRF variables from external validation sample
dep_obs <- VAMC$msefct1
sub_obs <- VAMC$msefct4
tht_obs <- VAMC$msefct5
pai_obs <- VAMC$msefct8

# Get predicted values for each regression model using external validation data:

# OLS multiple regression (RC scales)

```



```

dep_pred_lm <- predict.train(dep_lm_train, VAMC)
sub_pred_lm <- predict.train(sub_lm_train, VAMC)
tht_pred_lm <- predict.train(tht_lm_train, VAMC)
pai_pred_lm <- predict.train(pai_lm_train, VAMC)

# Bivariate regression
dep_pred_bi <- predict.train(dep_nb_train, VAMC)
sub_pred_bi <- predict.train(sub_nb_train, VAMC)
tht_pred_bi <- predict.train(tht_nb_train, VAMC)
pai_pred_bi <- predict.train(pai_nb_train, VAMC)

# Negative binomial regression
dep_pred_nb <- predict.train(dep_nb_train, VAMC)
sub_pred_nb <- predict.train(sub_nb_train, VAMC)
tht_pred_nb <- predict.train(tht_nb_train, VAMC)
pai_pred_nb <- predict.train(pai_nb_train, VAMC)

# Ridge regression (RC scales)
preds <- c("rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8",
"rfRC9")
test_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()
dep_pred_ridge <- predict(dep_ridge_train, s = dep_ridge_train$lambda.min, newx =
test_mat)
sub_pred_ridge <- predict(sub_ridge_train, s = sub_ridge_train$lambda.min, newx =
test_mat)
tht_pred_ridge <- predict(tht_ridge_train, s = tht_ridge_train$lambda.min, newx =
test_mat)
pai_pred_ridge <- predict(pai_ridge_train, s = pai_ridge_train$lambda.min, newx =
test_mat)

# Random forests regression (RC Scales)
dep_pred_rf <- predict.train(dep_rf_train, VAMC)
sub_pred_rf <- predict.train(sub_rf_train, VAMC)
tht_pred_rf <- predict.train(tht_rf_train, VAMC)
pai_pred_rf <- predict.train(pai_rf_train, VAMC)

# SVR (RC scales)
dep_pred_svr <- predict.train(dep_svr_train, VAMC)
sub_pred_svr <- predict.train(sub_svr_train, VAMC)
tht_pred_svr <- predict.train(tht_svr_train, VAMC)
pai_pred_svr <- predict.train(pai_svr_train, VAMC)

# OLS multiple regression (all scales)
dep_pred_lm_full <- predict.train(full_dep_reg_lm, VAMC)
sub_pred_lm_full <- predict.train(full_sub_reg_lm, VAMC)
tht_pred_lm_full <- predict.train(full_tht_reg_lm, VAMC)
pai_pred_lm_full <- predict.train(full_pai_reg_lm, VAMC)

# Ridge linear regression (all scales)
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
"rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
"rfCOG",
"rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
"rfMSF",
"rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
"rfDSF",
"rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
test_mat <- dplyr::select(VAMC, one_of(preds)) %>%
  as.matrix()
dep_pred_ridge_full <- predict(full_dep_ridge_reg, s = full_dep_ridge_reg$lambda.min,
newx = test_mat)

```

```

sub_pred_ridge_full <- predict(full_sub_ridge_reg, s = full_sub_ridge_reg$lambda.min,
newx = test_mat)
tht_pred_ridge_full <- predict(full_tht_ridge_reg, s = full_tht_ridge_reg$lambda.min,
newx = test_mat)
pai_pred_ridge_full <- predict(full_pai_ridge_reg, s = full_pai_ridge_reg$lambda.min,
newx = test_mat)

# Manual selection regression
dep_pred_man <- predict.train(man_dep_reg_lm, VAMC)
sub_pred_man <- predict.train(man_sub_reg_lm, VAMC)
tht_pred_man <- predict.train(man_tht_reg_lm, VAMC)
pai_pred_man <- predict.train(man_pai_reg_lm, VAMC)

# Forward stepwise selection
dep_pred_fsw <- predict.train(dep_reg_fsw, VAMC)
sub_pred_fsw <- predict.train(sub_reg_fsw, VAMC)
tht_pred_fsw <- predict.train(tht_reg_fsw, VAMC)
pai_pred_fsw <- predict.train(pai_reg_fsw, VAMC)

# Backward stepwise deletion
dep_pred_bsw <- predict.train(dep_reg_bsw, VAMC)
sub_pred_bsw <- predict.train(sub_reg_bsw, VAMC)
tht_pred_bsw <- predict.train(tht_reg_bsw, VAMC)
pai_pred_bsw <- predict.train(pai_reg_bsw, VAMC)

# Hybrid stepwise selection
dep_pred_hsw <- predict(dep_reg_hsw, VAMC)
sub_pred_hsw <- predict(sub_reg_hsw, VAMC)
tht_pred_hsw <- predict(tht_reg_hsw, VAMC)
pai_pred_hsw <- predict(pai_reg_hsw, VAMC)

# Lasso regression
preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
"rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC", "rfNUC",
"rfCOG",
"rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP", "rfBRF",
"rfMSF",
"rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV", "rfSHY",
"rfDSF",
"rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
train_mat <- dplyr::select(HCMC, one_of(preds)) %>%
  as.matrix()
dep_pred_lasso_full <- predict(full_dep_lasso_reg, s = full_dep_lasso_reg$lambda.min,
newx = test_mat)
sub_pred_lasso_full <- predict(full_sub_lasso_reg, s = full_sub_lasso_reg$lambda.min,
newx = test_mat)
tht_pred_lasso_full <- predict(full_tht_lasso_reg, s = full_tht_lasso_reg$lambda.min,
newx = test_mat)
pai_pred_lasso_full <- predict(full_pai_lasso_reg, s = full_pai_lasso_reg$lambda.min,
newx = test_mat)

# Random forests (all scales)
dep_pred_rf_full <- predict.train(dep_rf_train, VAMC)
sub_pred_rf_full <- predict.train(sub_rf_train, VAMC)
tht_pred_rf_full <- predict.train(tht_rf_train, VAMC)
pai_pred_rf_full <- predict.train(pai_rf_train, VAMC)

# Support vector regression (all scales)
dep_pred_svr_full <- predict.train(full_dep_svr_train, VAMC)
sub_pred_svr_full <- predict.train(full_sub_svr_train, VAMC)
tht_pred_svr_full <- predict.train(full_tht_svr_train, VAMC)
pai_pred_svr_full <- predict.train(full_pai_svr_train, VAMC)

```

```

# Generate observed vs. predicted fit plots
dep_lm_fit <- fit_plot(dep_obs, dep_pred_lm, "OLS Linear Regression (RC Scales)")
dep_bi_fit <- fit_plot(dep_obs, dep_pred_bi, "Bivariate Regression")
dep_nb_fit <- fit_plot(dep_obs, dep_pred_nb, "Negative Binomial Regression")
dep_ridge_fit <- fit_plot(dep_obs, dep_pred_ridge, "Ridge Regression (RC Scales)")
dep_rf_fit <- fit_plot(dep_obs, dep_pred_rf, "Random Forests Regression (RC Scales)")
dep_svr_fit <- fit_plot(dep_obs, dep_pred_svr, "SVR (RC Scales)")
dep_lm_full_fit <- fit_plot(dep_obs, dep_pred_lm_full, "OLS Linear Regression (all
Scales)")
dep_ridge_full_fit <- fit_plot(dep_obs, dep_pred_ridge_full, "Ridge Regression (all
scales)")
dep_man_fit <- fit_plot(dep_obs, dep_pred_man, "Manual Selection")
dep_fsw_fit <- fit_plot(dep_obs, dep_pred_fsw, "Forward Selection")
dep_bsw_fit <- fit_plot(dep_obs, dep_pred_bsw, "Backward Deletion")
dep_hsw_fit <- fit_plot(dep_obs, dep_pred_hsw, "Forward-Backward Stepwise Selection")
dep_lasso_fit <- fit_plot(dep_obs, dep_pred_lasso_full, "Lasso")
dep_rf_full_fit <- fit_plot(dep_obs, dep_pred_rf_full, "Random Forests Regression (all
scales)")
dep_svr_full_fit <- fit_plot(dep_obs, dep_pred_svr_full, "SVR (all Scales)")

sub_lm_fit <- fit_plot(sub_obs, sub_pred_lm, "OLS Linear Regression (RC Scales)")
sub_bi_fit <- fit_plot(sub_obs, sub_pred_bi, "Bivariate Regression")
sub_nb_fit <- fit_plot(sub_obs, sub_pred_nb, "Negative Binomial Regression")
sub_ridge_fit <- fit_plot(sub_obs, sub_pred_ridge, "Ridge Regression (RC Scales)")
sub_rf_fit <- fit_plot(sub_obs, sub_pred_rf, "Random Forests Regression (RC Scales)")
sub_svr_fit <- fit_plot(sub_obs, sub_pred_svr, "SVR (RC Scales)")
sub_lm_full_fit <- fit_plot(sub_obs, sub_pred_lm_full, "OLS Linear Regression (all
Scales)")
sub_ridge_full_fit <- fit_plot(sub_obs, sub_pred_ridge_full, "Ridge Regression (all
scales)")
sub_man_fit <- fit_plot(sub_obs, sub_pred_man, "Manual Selection")
sub_fsw_fit <- fit_plot(sub_obs, sub_pred_fsw, "Forward Selection")
sub_bsw_fit <- fit_plot(sub_obs, sub_pred_bsw, "Backward Deletion")
sub_hsw_fit <- fit_plot(sub_obs, sub_pred_hsw, "Forward-Backward Stepwise Selection")
sub_lasso_fit <- fit_plot(sub_obs, sub_pred_lasso_full, "Lasso")
sub_rf_full_fit <- fit_plot(sub_obs, sub_pred_rf_full, "Random Forests Regression (all
scales)")
sub_svr_full_fit <- fit_plot(sub_obs, sub_pred_svr_full, "SVR (all Scales)")

tht_lm_fit <- fit_plot(tht_obs, tht_pred_lm, "OLS Linear Regression (RC Scales)")
tht_bi_fit <- fit_plot(tht_obs, tht_pred_bi, "Bivariate Regression")
tht_nb_fit <- fit_plot(tht_obs, tht_pred_nb, "Negative Binomial Regression")
tht_ridge_fit <- fit_plot(tht_obs, tht_pred_ridge, "Ridge Regression (RC Scales)")
tht_rf_fit <- fit_plot(tht_obs, tht_pred_rf, "Random Forests Regression (RC Scales)")
tht_svr_fit <- fit_plot(tht_obs, tht_pred_svr, "SVR (RC Scales)")
tht_lm_full_fit <- fit_plot(tht_obs, tht_pred_lm_full, "OLS Linear Regression (all
Scales)")
tht_ridge_full_fit <- fit_plot(tht_obs, tht_pred_ridge_full, "Ridge Regression (all
scales)")
tht_man_fit <- fit_plot(tht_obs, tht_pred_man, "Manual Selection")
tht_fsw_fit <- fit_plot(tht_obs, tht_pred_fsw, "Forward Selection")
tht_bsw_fit <- fit_plot(tht_obs, tht_pred_bsw, "Backward Deletion")
tht_hsw_fit <- fit_plot(tht_obs, tht_pred_hsw, "Forward-Backward Stepwise Selection")
tht_lasso_fit <- fit_plot(tht_obs, tht_pred_lasso_full, "Lasso")
tht_rf_full_fit <- fit_plot(tht_obs, tht_pred_rf_full, "Random Forests Regression (all
scales)")
tht_svr_full_fit <- fit_plot(tht_obs, tht_pred_svr_full, "SVR (all Scales)")

pai_lm_fit <- fit_plot(pai_obs, pai_pred_lm, "OLS Linear Regression (RC Scales)")
pai_bi_fit <- fit_plot(pai_obs, pai_pred_bi, "Bivariate Regression")
pai_nb_fit <- fit_plot(pai_obs, pai_pred_nb, "Negative Binomial Regression")
pai_ridge_fit <- fit_plot(pai_obs, pai_pred_ridge, "Ridge Regression (RC Scales)")
pai_rf_fit <- fit_plot(pai_obs, pai_pred_rf, "Random Forests Regression (RC Scales)")

```

```

pai_svr_fit <- fit_plot(pai_obs, pai_pred_svr, "SVR (RC Scales)")
pai_lm_full_fit <- fit_plot(pai_obs, pai_pred_lm_full, "OLS Linear Regression (all
Scales)")
pai_ridge_full_fit <- fit_plot(pai_obs, pai_pred_ridge_full, "Ridge Regression (all
scales)")
pai_man_fit <- fit_plot(pai_obs, pai_pred_man, "Manual Selection")
pai_fsw_fit <- fit_plot(pai_obs, pai_pred_fsw, "Forward Selection")
pai_bsw_fit <- fit_plot(pai_obs, pai_pred_bsw, "Backward Deletion")
pai_hsw_fit <- fit_plot(pai_obs, pai_pred_hsw, "Forward-Backward Stepwise Selection")
pai_lasso_fit <- fit_plot(pai_obs, pai_pred_lasso_full, "Lasso")
pai_rf_full_fit <- fit_plot(pai_obs, pai_pred_rf_full, "Random Forests Regression (all
scales)")
pai_svr_full_fit <- fit_plot(pai_obs, pai_pred_svr_full, "SVR (all Scales)")

# Generate residual scatter plots
dep_lm_resid <- resid_plot(dep_obs, dep_pred_lm, "OLS (RC Scales)")
dep_bi_resid <- resid_plot(dep_obs, dep_pred_bi, "Single Predictor")
dep_nb_resid <- resid_plot(dep_obs, dep_pred_nb, "Negative Binomial (RC Scales)")
dep_ridge_resid <- resid_plot(dep_obs, dep_pred_ridge, "Ridge Regression (RC Scales)")
dep_rf_resid <- resid_plot(dep_obs, dep_pred_rf, "Random Forests Regression (RC
Scales)")
dep_svr_resid <- resid_plot(dep_obs, dep_pred_svr, "SVR (RC Scales)")
dep_lm_full_resid <- resid_plot(dep_obs, dep_pred_lm_full, "OLS Linear Regression (all
Scales)")
dep_ridge_full_resid <- resid_plot(dep_obs, dep_pred_ridge_full, "Ridge Regression
(all scales)")
dep_man_resid <- resid_plot(dep_obs, dep_pred_man, "Manual Selection")
dep_fsw_resid <- resid_plot(dep_obs, dep_pred_fsw, "Forward Selection")
dep_bsw_resid <- resid_plot(dep_obs, dep_pred_bsw, "Backward Deletion")
dep_hsw_resid <- resid_plot(dep_obs, dep_pred_hsw, "Forward-Backward Stepwise
Selection")
dep_lasso_resid <- resid_plot(dep_obs, dep_pred_lasso_full, "Lasso")
dep_rf_full_resid <- resid_plot(dep_obs, dep_pred_rf_full, "Random Forests Regression
(all scales)")
dep_svr_full_resid <- resid_plot(dep_obs, dep_pred_svr_full, "SVR (all Scales)")

sub_lm_resid <- resid_plot(sub_obs, sub_pred_lm, "OLS Linear Regression (RC Scales)")
sub_bi_resid <- resid_plot(sub_obs, sub_pred_bi, "Bivariate Regression")
sub_nb_resid <- resid_plot(sub_obs, sub_pred_nb, "Negative Binomial Regression")
sub_ridge_resid <- resid_plot(sub_obs, sub_pred_ridge, "Ridge Regression (RC Scales)")
sub_rf_resid <- resid_plot(sub_obs, sub_pred_rf, "Random Forests Regression (RC
Scales)")
sub_svr_resid <- resid_plot(sub_obs, sub_pred_svr, "SVR (RC Scales)")
sub_lm_full_resid <- resid_plot(sub_obs, sub_pred_lm_full, "OLS Linear Regression (all
Scales)")
sub_ridge_full_resid <- resid_plot(sub_obs, sub_pred_ridge_full, "Ridge Regression
(all scales)")
sub_man_resid <- resid_plot(sub_obs, sub_pred_man, "Manual Selection")
sub_fsw_resid <- resid_plot(sub_obs, sub_pred_fsw, "Forward Selection")
sub_bsw_resid <- resid_plot(sub_obs, sub_pred_bsw, "Backward Deletion")
sub_hsw_resid <- resid_plot(sub_obs, sub_pred_hsw, "Forward-Backward Stepwise
Selection")
sub_lasso_resid <- resid_plot(sub_obs, sub_pred_lasso_full, "Lasso")
sub_rf_full_resid <- resid_plot(sub_obs, sub_pred_rf_full, "Random Forests Regression
(all scales)")
sub_svr_full_resid <- resid_plot(sub_obs, sub_pred_svr_full, "SVR (all Scales)")

tht_lm_resid <- resid_plot(tht_obs, tht_pred_lm, "OLS Linear Regression (RC Scales)")
tht_bi_resid <- resid_plot(tht_obs, tht_pred_bi, "Bivariate Regression")
tht_nb_resid <- resid_plot(tht_obs, tht_pred_nb, "Negative Binomial Regression")
tht_ridge_resid <- resid_plot(tht_obs, tht_pred_ridge, "Ridge Regression (RC Scales)")
tht_rf_resid <- resid_plot(tht_obs, tht_pred_rf, "Random Forests Regression (RC
Scales)")

```

```

tht_svr_resid <- resid_plot(tht_obs, tht_pred_svr, "SVR (RC Scales)")
tht_lm_full_resid <- resid_plot(tht_obs, tht_pred_lm_full, "OLS Linear Regression (all
Scales)")
tht_ridge_full_resid <- resid_plot(tht_obs, tht_pred_ridge_full, "Ridge Regression
(all scales)")
tht_man_resid <- resid_plot(tht_obs, tht_pred_man, "Manual Selection")
tht_fsw_resid <- resid_plot(tht_obs, tht_pred_fsw, "Forward Selection")
tht_bsw_resid <- resid_plot(tht_obs, tht_pred_bsw, "Backward Deletion")
tht_hsw_resid <- resid_plot(tht_obs, tht_pred_hsw, "Forward-Backward Stepwise
Selection")
tht_lasso_resid <- resid_plot(tht_obs, tht_pred_lasso_full, "Lasso")
tht_rf_full_resid <- resid_plot(tht_obs, tht_pred_rf_full, "Random Forests Regression
(all scales)")
tht_svr_full_resid <- resid_plot(tht_obs, tht_pred_svr_full, "SVR (all Scales)")

pai_lm_resid <- resid_plot(pai_obs, pai_pred_lm, "OLS Linear Regression (RC Scales)")
pai_bi_resid <- resid_plot(pai_obs, pai_pred_bi, "Bivariate Regression")
pai_nb_resid <- resid_plot(pai_obs, pai_pred_nb, "Negative Binomial Regression")
pai_ridge_resid <- resid_plot(pai_obs, pai_pred_ridge, "Ridge Regression (RC Scales)")
pai_rf_resid <- resid_plot(pai_obs, pai_pred_rf, "Random Forests Regression (RC
Scales)")
pai_svr_resid <- resid_plot(pai_obs, pai_pred_svr, "SVR (RC Scales)")
pai_lm_full_resid <- resid_plot(pai_obs, pai_pred_lm_full, "OLS Linear Regression (all
Scales)")
pai_ridge_full_resid <- resid_plot(pai_obs, pai_pred_ridge_full, "Ridge Regression
(all scales)")
pai_man_resid <- resid_plot(pai_obs, pai_pred_man, "Manual Selection")
pai_fsw_resid <- resid_plot(pai_obs, pai_pred_fsw, "Forward Selection")
pai_bsw_resid <- resid_plot(pai_obs, pai_pred_bsw, "Backward Deletion")
pai_hsw_resid <- resid_plot(pai_obs, pai_pred_hsw, "Forward-Backward Stepwise
Selection")
pai_lasso_resid <- resid_plot(pai_obs, pai_pred_lasso_full, "Lasso")
pai_rf_full_resid <- resid_plot(pai_obs, pai_pred_rf_full, "Random Forests Regression
(all scales)")
pai_svr_full_resid <- resid_plot(pai_obs, pai_pred_svr_full, "SVR (all Scales)")

# Output fit and residual plots
ppi <- 300
setwd(outwd)
png("Fits - Dep (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(dep_lm_fit, dep_bi_fit, dep_nb_fit, dep_ridge_fit, dep_rf_fit,
  dep_svr_fit, dep_lm_full_fit, dep_ridge_full_fit, dep_man_fit,
  dep_fsw_fit, dep_bsw_fit, dep_hsw_fit, dep_lasso_fit,
  dep_rf_full_fit, dep_svr_full_fit,
  ncol = 4, nrow = 4,
  top = textGrob("RRF Depression Fit Plots by Model", gp = gpar(fontsize =
16))))
dev.off()
png("Fits - Sub (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(sub_lm_fit, sub_bi_fit, sub_nb_fit, sub_ridge_fit, sub_rf_fit,
  sub_svr_fit, sub_lm_full_fit, sub_ridge_full_fit, sub_man_fit,
  sub_fsw_fit, sub_bsw_fit, sub_hsw_fit, sub_lasso_fit,
  sub_rf_full_fit, sub_svr_full_fit,
  ncol = 4, nrow = 4,
  top = textGrob("RRF Substance Abuse Fit Plots by Model", gp =
gpar(fontsize = 16))))
dev.off()
png("Fits - Del (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(tht_lm_fit, tht_bi_fit, tht_nb_fit, tht_ridge_fit, tht_rf_fit,
  tht_svr_fit, tht_lm_full_fit, tht_ridge_full_fit, tht_man_fit,
  tht_fsw_fit, tht_bsw_fit, tht_hsw_fit, tht_lasso_fit,
  tht_rf_full_fit, tht_svr_full_fit,
  ncol = 4, nrow = 4,

```

```

top = textGrob("RRF Delusions Fit Plots by Model", gp =
gpar(fontsize = 16)))
dev.off()
png("Fits - Pain (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(pai_lm_fit, pai_bi_fit, pai_nb_fit, pai_ridge_fit, pai_rf_fit,
pai_svr_fit, pai_lm_full_fit, pai_ridge_full_fit, pai_man_fit,
pai_fsw_fit, pai_bsw_fit, pai_hsw_fit, pai_lasso_fit,
pai_rf_full_fit, pai_svr_full_fit,
ncol = 4, nrow = 4,
top = textGrob("RRF Pain Fit Plots by Model", gp = gpar(fontsize =
16))))
dev.off()
png("Resids - Dep (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(dep_lm_resid, dep_bi_resid, dep_nb_resid, dep_ridge_resid,
dep_rf_resid,
dep_svr_resid, dep_lm_full_resid, dep_ridge_full_resid,
dep_man_resid,
dep_fsw_resid, dep_bsw_resid, dep_hsw_resid, dep_lasso_resid,
dep_rf_full_resid, dep_svr_full_resid,
ncol = 4, nrow = 4,
top = textGrob("RRF Depression Residual Plots by Model", gp =
gpar(fontsize = 16)))
dev.off()
png("Resids - Sub (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(sub_lm_resid, sub_bi_resid, sub_nb_resid, sub_ridge_resid,
sub_rf_resid,
sub_svr_resid, sub_lm_full_resid, sub_ridge_full_resid,
sub_man_resid,
sub_fsw_resid, sub_bsw_resid, sub_hsw_resid, sub_lasso_resid,
sub_rf_full_resid, sub_svr_full_resid,
ncol = 4, nrow = 4,
top = textGrob("RRF Substance Abuse Residual Plots by Model", gp =
gpar(fontsize = 16)))
dev.off()
png("Resids - Del (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(tht_lm_resid, tht_bi_resid, tht_nb_resid, tht_ridge_resid,
tht_rf_resid,
tht_svr_resid, tht_lm_full_resid, tht_ridge_full_resid,
tht_man_resid,
tht_fsw_resid, tht_bsw_resid, tht_hsw_resid, tht_lasso_resid,
tht_rf_full_resid, tht_svr_full_resid,
ncol = 4, nrow = 4,
top = textGrob("RRF Delusions Residual Plots by Model", gp =
gpar(fontsize = 16)))
dev.off()
png("Resids - Pain (VAMC).png", width = 18 * ppi, height = 12 * ppi, res = ppi)
print(grid.arrange(pai_lm_resid, pai_bi_resid, pai_nb_resid, pai_ridge_resid,
pai_rf_resid,
pai_svr_resid, pai_lm_full_resid, pai_ridge_full_resid,
pai_man_resid,
pai_fsw_resid, pai_bsw_resid, pai_hsw_resid, pai_lasso_resid,
pai_rf_full_resid, pai_svr_full_resid,
ncol = 4, nrow = 4,
top = textGrob("RRF Pain Residual Plots by Model", gp =
gpar(fontsize = 16)))
dev.off()
setwd(wd)

# Calibration plots for classification predictions
# -----

```

```

# Prepare predictor matrices for glmnet
RC_preds <- c("rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4", "rfRC6", "rfRC7", "rfRC8",
"rfRC9")
full_preds <- c("rfEID", "rfBXD", "rfTHD", "rfRCd", "rfRC1", "rfRC2", "rfRC3", "rfRC4",
"rfRC6", "rfRC7", "rfRC8", "rfRC9", "rfMLS", "rfGIC", "rfHPC",
"rfNUC", "rfCOG",
"rfSUI", "rfHLP", "rfSFD", "rfNFC", "rfSTW", "rfAXY", "rfANP",
"rfBRF", "rfMSF",
"rfJCP", "rfSUB", "rfAGG", "rfACT", "rfFML", "rfIPP", "rfSAV",
"rfSHY", "rfDSF",
"rfAES", "rfMEC", "rfAGGR", "rfPSYC", "rfDISC", "rfNEGE", "rfINTR")
RC_VAMC_mat <- dplyr::select(VAMC, one_of(RC_preds)) %>%
  as.matrix() # For glmnet predictions
RC_ppath_mat <- dplyr::select(ppath, one_of(RC_preds)) %>%
  as.matrix()
full_VAMC_mat <- dplyr::select(VAMC, one_of(full_preds)) %>%
  as.matrix() # For glmnet predictions
full_ppath_mat <- dplyr::select(ppath, one_of(full_preds)) %>%
  as.matrix()

# Calculate predicted class probabilities
dep_probs_lr_VAMC <- predict.train(dep_lr_train, VAMC, type = "prob")[,1]
dep_probs_blr_VAMC <- predict.train(dep_blr_train, VAMC, type = "prob")[,1]
dep_probs_rlr_VAMC <- 1 - predict(dep_rlr_train, s = dep_rlr_train$lambda.min, newx =
RC_VAMC_mat,
                                type = "response")
dep_probs_rfc_VAMC <- predict.train(dep_rfc_train, VAMC, type = "prob")[,1]
dep_probs_svm_VAMC <- predict.train(dep_svm_train, VAMC, type = "prob")[,1]
dep_probs_full_lr_VAMC <- predict.train(full_dep_class_lr, VAMC, type = "prob")[,1]
dep_probs_full_ridge_VAMC <- 1 - predict(full_dep_rlr_class, s =
full_dep_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
dep_probs_man_VAMC <- predict.train(man_dep_class_lr, VAMC, type = "prob")[,1]
dep_probs_fsw_VAMC <- predict.train(dep_class_fsw, VAMC, type = "prob")[,1]
dep_probs_bsw_VAMC <- predict.train(dep_class_bsw, VAMC, type = "prob")[,1]
dep_probs_hsw_VAMC <- 1 - predict(dep_class_hsw, VAMC, type = "response")
dep_probs_llr_VAMC <- 1 - predict(full_dep_llr_class, s =
full_dep_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
dep_probs_full_rfc_VAMC <- predict.train(full_dep_rfc_train, VAMC, type = "prob")[,1]
dep_probs_full_svm_VAMC <- predict.train(full_dep_svm_train, VAMC, type = "prob")[,1]

sub_probs_lr_VAMC <- predict.train(sub_lr_train, VAMC, type = "prob")[,1]
sub_probs_blr_VAMC <- predict.train(sub_blr_train, VAMC, type = "prob")[,1]
sub_probs_rlr_VAMC <- 1 - predict(sub_rlr_train, s = sub_rlr_train$lambda.min, newx =
RC_VAMC_mat,
                                type = "response")
sub_probs_rfc_VAMC <- predict.train(sub_rfc_train, VAMC, type = "prob")[,1]
sub_probs_svm_VAMC <- predict.train(sub_svm_train, VAMC, type = "prob")[,1]
sub_probs_full_lr_VAMC <- predict.train(full_sub_class_lr, VAMC, type = "prob")[,1]
sub_probs_full_ridge_VAMC <- 1 - predict(full_sub_rlr_class, s =
full_sub_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
sub_probs_man_VAMC <- predict.train(man_sub_class_lr, VAMC, type = "prob")[,1]
sub_probs_fsw_VAMC <- predict.train(sub_class_fsw, VAMC, type = "prob")[,1]
sub_probs_bsw_VAMC <- predict.train(sub_class_bsw, VAMC, type = "prob")[,1]
sub_probs_hsw_VAMC <- 1 - predict(sub_class_hsw, VAMC, type = "response")
sub_probs_llr_VAMC <- 1 - predict(full_sub_llr_class, s =
full_sub_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
sub_probs_full_rfc_VAMC <- predict.train(full_sub_rfc_train, VAMC, type = "prob")[,1]
sub_probs_full_svm_VAMC <- predict.train(full_sub_svm_train, VAMC, type = "prob")[,1]

```

```

tht_probs_lr_VAMC <- predict.train(tht_lr_train, VAMC, type = "prob")[,1]
tht_probs_blr_VAMC <- predict.train(tht_blr_train, VAMC, type = "prob")[,1]
tht_probs_rlr_VAMC <- 1 - predict(tht_rlr_train, s = tht_rlr_train$lambda.min, newx =
RC_VAMC_mat,
                                type = "response")
tht_probs_rfc_VAMC <- predict.train(tht_rfc_train, VAMC, type = "prob")[,1]
tht_probs_svm_VAMC <- predict.train(tht_svm_train, VAMC, type = "prob")[,1]
tht_probs_full_lr_VAMC <- predict.train(full_tht_class_lr, VAMC, type = "prob")[,1]
tht_probs_full_ridge_VAMC <- 1 - predict(full_tht_rlr_class, s =
full_tht_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
tht_probs_man_VAMC <- predict.train(man_tht_class_lr, VAMC, type = "prob")[,1]
tht_probs_fsw_VAMC <- predict.train(tht_class_fsw, VAMC, type = "prob")[,1]
tht_probs_bsw_VAMC <- predict.train(tht_class_bsw, VAMC, type = "prob")[,1]
tht_probs_hsw_VAMC <- 1 - predict(tht_class_hsw, VAMC, type = "response")
tht_probs_llr_VAMC <- 1 - predict(full_tht_llr_class, s =
full_tht_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
tht_probs_full_rfc_VAMC <- predict.train(full_tht_rfc_train, VAMC, type = "prob")[,1]
tht_probs_full_svm_VAMC <- predict.train(full_tht_svm_train, VAMC, type = "prob")[,1]

som_probs_lr_VAMC <- predict.train(som_lr_train, VAMC, type = "prob")[,1]
som_probs_blr_VAMC <- predict.train(som_blr_train, VAMC, type = "prob")[,1]
som_probs_rlr_VAMC <- 1 - predict(som_rlr_train, s = som_rlr_train$lambda.min, newx =
RC_VAMC_mat,
                                type = "response")
som_probs_rfc_VAMC <- predict.train(som_rfc_train, VAMC, type = "prob")[,1]
som_probs_svm_VAMC <- predict.train(som_svm_train, VAMC, type = "prob")[,1]
som_probs_full_lr_VAMC <- predict.train(full_som_class_lr, VAMC, type = "prob")[,1]
som_probs_full_ridge_VAMC <- 1 - predict(full_som_rlr_class, s =
full_som_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
som_probs_man_VAMC <- predict.train(man_som_class_lr, VAMC, type = "prob")[,1]
som_probs_fsw_VAMC <- predict.train(som_class_fsw, VAMC, type = "prob")[,1]
som_probs_bsw_VAMC <- predict.train(som_class_bsw, VAMC, type = "prob")[,1]
som_probs_hsw_VAMC <- 1 - predict(som_class_hsw, VAMC, type = "response")
som_probs_llr_VAMC <- 1 - predict(full_som_llr_class, s =
full_som_rlr_class$lambda.min, newx = full_VAMC_mat,
                                type = "response")
som_probs_full_rfc_VAMC <- predict.train(full_som_rfc_train, VAMC, type = "prob")[,1]
som_probs_full_svm_VAMC <- predict.train(full_som_svm_train, VAMC, type = "prob")[,1]

dep_probs_lr_ppath <- predict.train(dep_lr_train, ppath, type = "prob")[,1]
dep_probs_blr_ppath <- predict.train(dep_blr_train, ppath, type = "prob")[,1]
dep_probs_rlr_ppath <- 1 - predict(dep_rlr_train, s = dep_rlr_train$lambda.min, newx =
RC_ppath_mat,
                                type = "response")
dep_probs_rfc_ppath <- predict.train(dep_rfc_train, ppath, type = "prob")[,1]
dep_probs_svm_ppath <- predict.train(dep_svm_train, ppath, type = "prob")[,1]
dep_probs_full_lr_ppath <- predict.train(full_dep_class_lr, ppath, type = "prob")[,1]
dep_probs_full_ridge_ppath <- 1 - predict(full_dep_rlr_class, s =
full_dep_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
dep_probs_man_ppath <- predict.train(man_dep_class_lr, ppath, type = "prob")[,1]
dep_probs_fsw_ppath <- predict.train(dep_class_fsw, ppath, type = "prob")[,1]
dep_probs_bsw_ppath <- predict.train(dep_class_bsw, ppath, type = "prob")[,1]
dep_probs_hsw_ppath <- 1 - predict(dep_class_hsw, ppath, type = "response")
dep_probs_llr_ppath <- 1 - predict(full_dep_llr_class, s =
full_dep_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
dep_probs_full_rfc_ppath <- predict.train(full_dep_rfc_train, ppath, type =
"prob")[,1]

```



```

dep_probs_full_svm_ppath <- predict.train(full_dep_svm_train, ppath, type =
"prob")[,1]

sub_probs_lr_ppath <- predict.train(sub_lr_train, ppath, type = "prob")[,1]
sub_probs_blr_ppath <- predict.train(sub_blr_train, ppath, type = "prob")[,1]
sub_probs_rlr_ppath <- 1 - predict(sub_rlr_train, s = sub_rlr_train$lambda.min, newx =
RC_ppath_mat,
                                type = "response")
sub_probs_rfc_ppath <- predict.train(sub_rfc_train, ppath, type = "prob")[,1]
sub_probs_svm_ppath <- predict.train(sub_svm_train, ppath, type = "prob")[,1]
sub_probs_full_lr_ppath <- predict.train(full_sub_class_lr, ppath, type = "prob")[,1]
sub_probs_full_ridge_ppath <- 1 - predict(full_sub_rlr_class, s =
full_sub_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
sub_probs_man_ppath <- predict.train(man_sub_class_lr, ppath, type = "prob")[,1]
sub_probs_fsw_ppath <- predict.train(sub_class_fsw, ppath, type = "prob")[,1]
sub_probs_bsw_ppath <- predict.train(sub_class_bsw, ppath, type = "prob")[,1]
sub_probs_hsw_ppath <- 1 - predict(sub_class_hsw, ppath, type = "response")
sub_probs_llr_ppath <- 1 - predict(full_sub_llr_class, s =
full_sub_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
sub_probs_full_rfc_ppath <- predict.train(full_sub_rfc_train, ppath, type =
"prob")[,1]
sub_probs_full_svm_ppath <- predict.train(full_sub_svm_train, ppath, type =
"prob")[,1]

tht_probs_lr_ppath <- predict.train(tht_lr_train, ppath, type = "prob")[,1]
tht_probs_blr_ppath <- predict.train(tht_blr_train, ppath, type = "prob")[,1]
tht_probs_rlr_ppath <- 1 - predict(tht_rlr_train, s = tht_rlr_train$lambda.min, newx =
RC_ppath_mat,
                                type = "response")
tht_probs_rfc_ppath <- predict.train(tht_rfc_train, ppath, type = "prob")[,1]
tht_probs_svm_ppath <- predict.train(tht_svm_train, ppath, type = "prob")[,1]
tht_probs_full_lr_ppath <- predict.train(full_tht_class_lr, ppath, type = "prob")[,1]
tht_probs_full_ridge_ppath <- 1 - predict(full_tht_rlr_class, s =
full_tht_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
tht_probs_man_ppath <- predict.train(man_tht_class_lr, ppath, type = "prob")[,1]
tht_probs_fsw_ppath <- predict.train(tht_class_fsw, ppath, type = "prob")[,1]
tht_probs_bsw_ppath <- predict.train(tht_class_bsw, ppath, type = "prob")[,1]
tht_probs_hsw_ppath <- 1 - predict(tht_class_hsw, ppath, type = "response")
tht_probs_llr_ppath <- 1 - predict(full_tht_llr_class, s =
full_tht_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
tht_probs_full_rfc_ppath <- predict.train(full_tht_rfc_train, ppath, type =
"prob")[,1]
tht_probs_full_svm_ppath <- predict.train(full_tht_svm_train, ppath, type =
"prob")[,1]

som_probs_lr_ppath <- predict.train(som_lr_train, ppath, type = "prob")[,1]
som_probs_blr_ppath <- predict.train(som_blr_train, ppath, type = "prob")[,1]
som_probs_rlr_ppath <- 1 - predict(som_rlr_train, s = som_rlr_train$lambda.min, newx =
RC_ppath_mat,
                                type = "response")
som_probs_rfc_ppath <- predict.train(som_rfc_train, ppath, type = "prob")[,1]
som_probs_svm_ppath <- predict.train(som_svm_train, ppath, type = "prob")[,1]
som_probs_full_lr_ppath <- predict.train(full_som_class_lr, ppath, type = "prob")[,1]
som_probs_full_ridge_ppath <- 1 - predict(full_som_rlr_class, s =
full_som_rlr_class$lambda.min, newx = full_ppath_mat,
                                type = "response")
som_probs_man_ppath <- predict.train(man_som_class_lr, ppath, type = "prob")[,1]
som_probs_fsw_ppath <- predict.train(som_class_fsw, ppath, type = "prob")[,1]
som_probs_bsw_ppath <- predict.train(som_class_bsw, ppath, type = "prob")[,1]

```

```

som_probs_hsw_ppath <- 1 - predict(som_class_hsw, ppath, type = "response")
som_probs_llr_ppath <- 1 - predict(full_som_llr_class, s =
full_som_rlr_class$lambda.min, newx = full_ppath_mat,
type = "response")
som_probs_full_rfc_ppath <- predict.train(full_som_rfc_train, ppath, type =
"prob")[,1]
som_probs_full_svm_ppath <- predict.train(full_som_svm_train, ppath, type =
"prob")[,1]

# Generate calibration curves
dep_VAMC_cal_RC <- calibration(dep_do ~ dep_probs_lr_VAMC + dep_probs_rlr_VAMC +
dep_probs_rfc_VAMC + dep_probs_svm_VAMC,
data = VAMC)
dep_VAMC_cal_full <- calibration(dep_do ~ dep_probs_full_lr_VAMC + dep_probs_blr_VAMC
+
dep_probs_full_ridge_VAMC + dep_probs_man_VAMC +
dep_probs_fsw_VAMC +
dep_probs_bsw_VAMC + dep_probs_hsw_VAMC +
dep_probs_llr_VAMC +
dep_probs_full_rfc_VAMC + dep_probs_full_svm_VAMC,
data = VAMC)
sub_VAMC_cal_RC <- calibration(sub_do ~ sub_probs_lr_VAMC + sub_probs_rlr_VAMC +
sub_probs_rfc_VAMC + sub_probs_svm_VAMC,
data = VAMC)
sub_VAMC_cal_full <- calibration(sub_do ~ sub_probs_full_lr_VAMC + sub_probs_blr_VAMC
+
sub_probs_full_ridge_VAMC + sub_probs_man_VAMC +
sub_probs_fsw_VAMC +
sub_probs_bsw_VAMC + sub_probs_hsw_VAMC +
sub_probs_llr_VAMC +
sub_probs_full_rfc_VAMC + sub_probs_full_svm_VAMC,
data = VAMC)
tht_VAMC_cal_RC <- calibration(thought_do ~ tht_probs_lr_VAMC + tht_probs_rlr_VAMC +
tht_probs_rfc_VAMC + tht_probs_svm_VAMC,
data = VAMC)
tht_VAMC_cal_full <- calibration(thought_do ~ tht_probs_full_lr_VAMC +
tht_probs_blr_VAMC +
tht_probs_full_ridge_VAMC + tht_probs_man_VAMC +
tht_probs_fsw_VAMC +
tht_probs_bsw_VAMC + tht_probs_hsw_VAMC +
tht_probs_llr_VAMC +
tht_probs_full_rfc_VAMC + tht_probs_full_svm_VAMC,
data = VAMC)
som_VAMC_cal_RC <- calibration(som_do ~ som_probs_lr_VAMC + som_probs_rlr_VAMC +
som_probs_rfc_VAMC + som_probs_svm_VAMC,
data = VAMC)
som_VAMC_cal_full <- calibration(som_do ~ som_probs_full_lr_VAMC + som_probs_blr_VAMC
+
som_probs_full_ridge_VAMC + som_probs_man_VAMC +
som_probs_fsw_VAMC +
som_probs_bsw_VAMC + som_probs_hsw_VAMC +
som_probs_llr_VAMC +
som_probs_full_rfc_VAMC + som_probs_full_svm_VAMC,
data = VAMC)
dep_ppath_cal_RC <- calibration(dep_do ~ dep_probs_lr_ppath + dep_probs_rlr_ppath +
dep_probs_rfc_ppath + dep_probs_svm_ppath,
data = ppath)
dep_ppath_cal_full <- calibration(dep_do ~ dep_probs_full_lr_ppath +
dep_probs_blr_ppath +
dep_probs_full_ridge_ppath + dep_probs_man_ppath +
dep_probs_fsw_ppath +

```

```

dep_probs_llr_ppath +
dep_probs_full_svm_ppath,
data = ppath)
sub_ppath_cal_RC <- calibration(sub_do ~ sub_probs_lr_ppath + sub_probs_rlr_ppath +
sub_probs_rfc_ppath + sub_probs_svm_ppath,
data = ppath)
sub_ppath_cal_full <- calibration(sub_do ~ sub_probs_full_lr_ppath +
sub_probs_blr_ppath +
sub_probs_full_ridge_ppath + sub_probs_man_ppath +
sub_probs_fsw_ppath +
sub_probs_bsw_ppath + sub_probs_hsw_ppath +
sub_probs_llr_ppath +
sub_probs_full_rfc_ppath +
sub_probs_full_svm_ppath,
data = ppath)
tth_ppath_cal_RC <- calibration(thought_do ~ tth_probs_lr_ppath + tth_probs_rlr_ppath
+
tth_probs_rfc_ppath + tth_probs_svm_ppath,
data = ppath)
tth_ppath_cal_full <- calibration(thought_do ~ tth_probs_full_lr_ppath +
tth_probs_blr_ppath +
tth_probs_full_ridge_ppath + tth_probs_man_ppath +
tth_probs_fsw_ppath +
tth_probs_bsw_ppath + tth_probs_hsw_ppath +
tth_probs_llr_ppath +
tth_probs_full_rfc_ppath +
tth_probs_full_svm_ppath,
data = ppath)
som_ppath_cal_RC <- calibration(som_do ~ som_probs_lr_ppath + som_probs_rlr_ppath +
som_probs_rfc_ppath + som_probs_svm_ppath,
data = ppath)
som_ppath_cal_full <- calibration(som_do ~ som_probs_full_lr_ppath +
som_probs_blr_ppath +
som_probs_full_ridge_ppath + som_probs_man_ppath +
som_probs_fsw_ppath +
som_probs_bsw_ppath + som_probs_hsw_ppath +
som_probs_llr_ppath +
som_probs_full_rfc_ppath +
som_probs_full_svm_ppath,
data = ppath)

RC_probNames <- c("MLE Logistic", "Ridge Logistic", "Random Forests", "SVM")
full_probNames <- c("MLE (all scales)", "Bivariate", "Ridge (all scales)", "Manual
Selection",
"Forward Selection", "Backward Deletion", "Forward-Backward
Stepwise", "Lasso",
"Random Forests (all scales)", "SVM (all scales)")

dep_VAMC_cal_RC$probNames <- sub_VAMC_cal_RC$probNames <- tth_VAMC_cal_RC$probNames <-
som_ppath_cal_RC$probNames <- dep_ppath_cal_RC$probNames <-
sub_ppath_cal_RC$probNames <- tth_ppath_cal_RC$probNames <-
som_ppath_cal_RC$probNames <- RC_probNames

dep_VAMC_cal_full$probNames <- sub_VAMC_cal_full$probNames <-
tth_VAMC_cal_full$probNames <-
som_ppath_cal_full$probNames <- dep_ppath_cal_full$probNames <-
sub_ppath_cal_full$probNames <- tth_ppath_cal_full$probNames <-
som_ppath_cal_full$probNames <- full_probNames

# Plot curves

```

```

dep_VAMC_RC_calplot <- ggplot(dep_VAMC_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Depressive Disorders (RC SScale Models)")
sub_VAMC_RC_calplot <- ggplot(sub_VAMC_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Substance Use Disorders (RC SScale Models)")
tht_VAMC_RC_calplot <- ggplot(tht_VAMC_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Thought Disorders (RC SScale Models)")
som_VAMC_RC_calplot <- ggplot(som_VAMC_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Somatic Disorders (RC SScale Models)")
dep_VAMC_full_calplot <- ggplot(dep_VAMC_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Depressive Disorders (Feature Selection Models)")
sub_VAMC_full_calplot <- ggplot(sub_VAMC_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Substance Use Disorders (Feature Selection Models)")
tht_VAMC_full_calplot <- ggplot(tht_VAMC_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Thought Disorders (Feature Selection Models)")
som_VAMC_full_calplot <- ggplot(som_VAMC_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Somatic Disorders (Feature Selection Models)")

dep_ppath_RC_calplot <- ggplot(dep_ppath_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Depressive Disorders (RC SScale Models)")
sub_ppath_RC_calplot <- ggplot(sub_ppath_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Substance Use Disorders (RC SScale Models)")
tht_ppath_RC_calplot <- ggplot(tht_ppath_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Thought Disorders (RC SScale Models)")
som_ppath_RC_calplot <- ggplot(som_ppath_cal_RC, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = RC_probNames) +
  ggtitle("Somatic Disorders (RC SScale Models)")
dep_ppath_full_calplot <- ggplot(dep_ppath_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Depressive Disorders (Feature Selection Models)")
sub_ppath_full_calplot <- ggplot(sub_ppath_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Substance Use Disorders (Feature Selection Models)")
tht_ppath_full_calplot <- ggplot(tht_ppath_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +
  scale_color_hue(labels = full_probNames) +
  ggtitle("Thought Disorders (Feature Selection Models)")
som_ppath_full_calplot <- ggplot(som_ppath_cal_full, bwdith = 0, dwidth = 0) +
  geom_line() +

```

```

scale_color_hue(labels = full_probNames) +
ggtitle("Somatic Disorders (Feature Selection Models)")

# Summary statistics for RRF variables
mean(HCMC_reg_out$`MSE-Depressed`)
mean(HCMC_reg_out$`MSE-SubAbuse`)
mean(HCMC_reg_out$`MSE-Delusions`)
mean(HCMC_reg_out$`MSE-Pain`)
sd(HCMC_reg_out$`MSE-Depressed`)
sd(HCMC_reg_out$`MSE-SubAbuse`)
sd(HCMC_reg_out$`MSE-Delusions`)
sd(HCMC_reg_out$`MSE-Pain`)

mean(VAMC_reg_out$`MSE-Depressed`)
mean(VAMC_reg_out$`MSE-SubAbuse`)
mean(VAMC_reg_out$`MSE-Delusions`)
mean(VAMC_reg_out$`MSE-Pain`)
sd(VAMC_reg_out$`MSE-Depressed`)
sd(VAMC_reg_out$`MSE-SubAbuse`)
sd(VAMC_reg_out$`MSE-Delusions`)
sd(VAMC_reg_out$`MSE-Pain`)

#=====#
# Post-hoc analyses      #
#=====#

# Full-Scale Random Forests Regression (1000 bootstrap replications)
#=====

# Train models using full training data -----

# Define random forests training control. This will allow the algorithm to explore a
# greater range of tuning parameters.
rf_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random")

set.seed(25101)
big_dep_rf_train <- train(form = full_dep_reg_form, data = HCMC,
                        trControl = rf_control, method = "rf",
                        tuneLength = 15, ntree = 1000)

set.seed(25102)
big_sub_rf_train <- train(form = full_sub_reg_form, data = HCMC,
                        trControl = rf_control, method = "rf",
                        tuneLength = 15, ntree = 1000)

set.seed(25103)
big_tht_rf_train <- train(form = full_tht_reg_form, data = HCMC,
                        trControl = rf_control, method = "rf",
                        tuneLength = 15, ntree = 1000)

set.seed(25104)
big_pai_rf_train <- train(form = full_pai_reg_form, data = HCMC,
                        trControl = rf_control, method = "rf",
                        tuneLength = 15, ntree = 1000)

# Export random forests regression results
big_rf_mdls <- list(big_dep_rf_train, big_sub_rf_train, big_tht_rf_train,
big_pai_rf_train)

big_rf_out <- matrix(NA,
                    nrow = length(big_rf_mdls[[1]]$finalModel$importance) + 3,
                    ncol = length(big_rf_mdls))
rownames(big_rf_out) <- c(rownames(big_rf_mdls[[1]]$finalModel$importance),
                        "RMSE-train", "RMSE-CV", "RMSE-EV")

```

```

colnames(big_rf_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(big_rf_mdls)){
  curr_mdl <- big_rf_mdls[[i]]
  importance <- curr_mdl$finalModel$importance
  n_coefs <- length(importance)

  best_mtry <- curr_mdl$bestTune %>% as.numeric()
  best_idx <- which(curr_mdl$results$mtry == best_mtry)
  best_mtry_RMSE <- curr_mdl$results$RMSE[best_idx]

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  rmse_train <- RMSE(predict(curr_mdl, HCMC), as.matrix(HCMC[, reg_outcomes[i]]))
  rmse_cv <- best_mtry_RMSE
  rmse_ev <- RMSE(test_pred, as.matrix(VAMC[, reg_outcomes[i]]))

  big_rf_out[1:n_coefs, i] <- importance
  big_rf_out[n_coefs + 1, i] <- rmse_train
  big_rf_out[n_coefs + 2, i] <- rmse_cv
  big_rf_out[n_coefs + 3, i] <- rmse_ev
}

# Export random forests regression results
setwd(outwd)
write.csv(big_rf_out, "Full-Scale Random Forests Regression Output (1000B).csv")
setwd(wd)

# Full-Scale Random Forests Regression (1500 bootstrap replications)
#=====

# Train models using full training data -----

# Define random forests training control. This will allow the algorithm to explore a
# greater range of tuning parameters.
rf_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random")

set.seed(25201)
bigger_dep_rf_train <- train(form = full_dep_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 1500)

set.seed(25202)
bigger_sub_rf_train <- train(form = full_sub_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 1500)

set.seed(25203)
bigger_tht_rf_train <- train(form = full_tht_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 1500)

set.seed(25204)
bigger_pai_rf_train <- train(form = full_pai_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 1500)

# Export random forests regression results
bigger_rf_mdls <- list(bigger_dep_rf_train, bigger_sub_rf_train, bigger_tht_rf_train,
bigger_pai_rf_train)

```

```

bigger_rf_out <- matrix(NA,
                        nrow = length(bigger_rf_mdls[[1]]$finalModel$importance) + 3,
                        ncol = length(bigger_rf_mdls))
rownames(bigger_rf_out) <- c(rownames(bigger_rf_mdls[[1]]$finalModel$importance),
                             "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(bigger_rf_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(bigger_rf_mdls)){
  curr_mdl <- bigger_rf_mdls[[i]]
  importance <- curr_mdl$finalModel$importance
  n_coefs <- length(importance)

  best_mtry <- curr_mdl$bestTune %>% as.numeric()
  best_idx <- which(curr_mdl$results$mtry == best_mtry)
  best_mtry_RMSE <- curr_mdl$results$RMSE[best_idx]

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  rmse_train <- RMSE(predict(curr_mdl, HCMC), as.matrix(HCMC[, reg_outcomes[i]]))
  rmse_cv <- best_mtry_RMSE
  rmse_ev <- RMSE(test_pred, as.matrix(VAMC[, reg_outcomes[i]]))

  bigger_rf_out[1:n_coefs, i] <- importance
  bigger_rf_out[n_coefs + 1, i] <- rmse_train
  bigger_rf_out[n_coefs + 2, i] <- rmse_cv
  bigger_rf_out[n_coefs + 3, i] <- rmse_ev
}

# Export random forests regression results
setwd(outwd)
write.csv(bigger_rf_out, "Full-Scale Random Forests Regression Output (1500B).csv")
setwd(wd)

# Full-Scale Random Forests Regression (2000 bootstrap replications)
#=====

# Train models using full training data -----

# Define random forests training control. This will allow the algorithm to explore a
# greater range of tuning parameters.
rf_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search =
"random")

set.seed(25301)
biggest_dep_rf_train <- train(form = full_dep_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 2000)

set.seed(25302)
biggest_sub_rf_train <- train(form = full_sub_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 2000)

set.seed(25303)
biggest_tht_rf_train <- train(form = full_tht_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",
                             tuneLength = 15, ntree = 2000)

set.seed(25304)
biggest_pai_rf_train <- train(form = full_pai_reg_form, data = HCMC,
                             trControl = rf_control, method = "rf",

```

```

tuneLength = 15, ntree = 2000)

# Export random forests regression results
biggest_rf_mdls <- list(biggest_dep_rf_train, biggest_sub_rf_train,
biggest_tht_rf_train, biggest_pai_rf_train)

biggest_rf_out <- matrix(NA,
                        nrow = length(biggest_rf_mdls[[1]]$finalModel$importance) + 3,
                        ncol = length(biggest_rf_mdls))
rownames(biggest_rf_out) <- c(rownames(biggest_rf_mdls[[1]]$finalModel$importance),
                             "RMSE-train", "RMSE-CV", "RMSE-EV")
colnames(biggest_rf_out) <- c("Depression", "Sub Abuse", "Delusions", "Pain")

reg_outcomes <- c("msefct1", "msefct4", "msefct5", "msefct8")

for(i in 1:length(biggest_rf_mdls)){
  curr_mdl <- biggest_rf_mdls[[i]]
  importance <- curr_mdl$finalModel$importance
  n_coefs <- length(importance)

  best_mtry <- curr_mdl$bestTune %>% as.numeric()
  best_idx <- which(curr_mdl$results$mtry == best_mtry)
  best_mtry_RMSE <- curr_mdl$results$RMSE[best_idx]

  train_outcome <- curr_mdl$trainingData$outcome
  test_pred <- predict.train(curr_mdl, VAMC)
  test_outcome <- VAMC[, reg_outcomes[i]]

  rmse_train <- RMSE(predict(curr_mdl, HCMC), as.matrix(HCMC[, reg_outcomes[i]]))
  rmse_cv <- best_mtry_RMSE
  rmse_ev <- RMSE(test_pred, as.matrix(VAMC[, reg_outcomes[i]]))

  biggest_rf_out[1:n_coefs, i] <- importance
  biggest_rf_out[n_coefs + 1, i] <- rmse_train
  biggest_rf_out[n_coefs + 2, i] <- rmse_cv
  biggest_rf_out[n_coefs + 3, i] <- rmse_ev
}

# Export random forests regression results
setwd(outwd)
write.csv(biggest_rf_out, "Full-Scale Random Forests Regression Output (2000B).csv")
setwd(wd)

```