A Tutorial on Supervised Machine Learning Variable Selection Methods for the Social and Health Sciences in R

Catherine M. Bain[1*], Dingjing Shi[1], Lauren E. Ethridge[1,2], Jordan E. Norris[1], & Jordan E. Loeffelman[1]

[1]Department of Psychology, University of Oklahoma, Norman, OK

[2]Department of Pediatrics, Section on Developmental and Behavioral Pediatrics, University of Oklahoma Health Sciences Center, Oklahoma City, OK

Correspondence concerning this article should be addressed to Catherine M. Bain, Department of Psychology, University of Oklahoma, 455 W. Lindsey Street, Dale Hall Tower, Room 705, Norman, OK 73019, United States. Email: cbain1@ou.edu

Keywords: Machine Learning; Random Forest; SVM; LASSO; Elastic Net; Variable Selection; Supervised Learning; Genetic Algorithm; Metaheuristic; R; Misophonia

Primary Abstract

With recent increases in the size of datasets currently available in the behavioral and health sciences, the need for efficient and effective variable selection techniques has increased. A plethora of techniques exist, yet only a few are used within the psychological sciences (e.g., stepwise regression, which is most common, the LASSO, and Elastic Net). The purpose of this tutorial is to increase awareness of the various variable selection methods available in the popular statistical software R, and guide researchers through how each method can be used to select variables in the context of classification using a recent survey-based assessment of misophonia. Specifically, readers will learn about how to implement and interpret results from the LASSO, Elastic Net, a penalized SVM classifier, an implementation of random forest, and the genetic algorithm. The associated code and data implemented in this tutorial are available on OSF to allow for a more interactive experience. This paper is written with the assumption that individuals have at least a basic understanding of R.

A Tutorial on Supervised Machine Learning Variable Selection Methods for the Social and Health Sciences in R

In the field of behavioral and health sciences, selecting the right variables for a model is a crucial step in working to understand how aspects of human behavior are related. For example, we might want to know how one's personality or other traits affect their engagement with a particular type of treatment or how the symptoms of a particular disorder may present. In many cases, researchers not only want to understand how these aspects (i.e., variables) are related to each other and to overarching constructs but may also want to use the variables to classify individuals into groups (e.g., diagnosing clinical disorders, determining participant compliance, etc.). The accuracy of these classifications or predictions is greatly influenced by which variables a researcher uses to create the classifications. For example, if a researcher is interested in diagnosing someone with depression, the accuracy of the diagnosis would suffer if relying solely on the presence of a depressed mood. However, if they use a variety of variables like depressed mood, loss of interest in activities, hours slept, and change in appetite or weight, their classification would be more accurate.

Essentially, researchers must think critically about building their classification model. A good model not only allows researchers to understand the interrelationships among variables, but also creates an accurate model in terms of predicted classes. Variable selection techniques can help researchers to identify and select informative variables to build these models. The use of variable selection techniques can lead to more accurate predictions, reduce the computational cost of creating the model, and improve the parsimony of the model by eliminating redundant and irrelevant variables. For example, variable selection techniques have been used to build models pertaining to identifying exposure-outcome associations [1] as well as predicting mortality rates [2,3], psychological strain in teachers [4], and nomophobia [5].

Behavioral researchers often turn to stepwise regression to perform variable selection. An APA PsychINFO database search for the term "stepwise regression" returned 222 peer-reviewed articles published in the last 3 years using stepwise regression for variable selection. Stepwise regression, however, has many severe limitations and statistical experts do not recommend its use in any context. These limitations include the inability to distinguish signal (i.e., true predictor variables) from noise [6–9], underestimation of *p*-values, and failure to replicate [10,11]. As such, many alternative variable selection algorithms have been proposed in the literature, but behavioral researchers have been slow to adopt these new methods in place of more traditional methods [12,13]. One potential reason for this delay may be the disconnect between methodological and applied behavioral researchers, as much methodological research is often inaccessible for applied researchers at first (e.g., complex techniques, lack of published code, or no tutorials). An APA PsychINFO database search for the term "variable selection" returned 253 papers published in quantitative methods journals in the last 20 years, indicating that methodological researchers are dedicated to developing better approaches to variable selection than stepwise regression. Of these publications, however, only one is a tutorial [14].

Given the clear gap in the popularity of variable selection methodological research and lack of tutorials on how to apply them, the field would benefit greatly from additional tutorials on variable selection techniques with demonstrations of how to apply them to psychological datasets. The goal of this paper is to provide a tutorial on five variable selection techniques freely available to researchers in R. We will introduce the Least Absolute Shrinkage and Selection Operator (LASSO), Elastic Net, a version of the genetic algorithm (GA), and implementations of Support Vector Machines (SVMs) and Random Forest that have been adapted to perform variable selection. The manuscript is organized as follows. The first section illustrates the importance of variable selection in machine learning and explains why each of the five methods method were selected. Then, a

motivating example is provided pertaining to diagnosis of misophonia. The dataset was collected from a psychology research pool and represents an excellent example of an average dataset available to many behavioral and health researchers [15]. Within this example, there are three major sections. The first discusses methods using a logistic regression model (i.e., LASSO, EN, and the GA), the second discusses SVM, and the third pertains to random forest. Each technique is introduced, the code necessary to implement each technique is provided, and each technique's associated strengths and weaknesses are discussed.

Variable Selection in Machine Learning

Variable selection is a fundamental step in the process of building robust and efficient machine learning models, and its importance cannot be overstated [16,17]. It serves as a critical mechanism for optimizing model performance and ensuring its reliability across various tasks and datasets. Variable selection is advantageous with any model (e.g., regression, structural equation modeling, etc.) because, as mentioned previously, it leads to more accurate predictions, reduces the computational cost of the model, and improves the parsimony of the model by eliminating redundant and irrelevant variables. However, there are additional advantages of variable selection when paired with machine learning models. First, variable selection helps to manage dimensionality problems (i.e., when a dataset contains more predictors than it does observations). Over the years, technology such as the invention of online data collection platforms like Prolific or the creation of mobile health apps has allowed researchers to collect more complex data from increasingly larger samples. As datasets grow in both size and complexity, the number of variables may also increase, leading to computational inefficiencies and reduced model interpretability [18]. By carefully selecting relevant variables, we can effectively reduce the dimensionality of the data, thereby streamlining the computational process and facilitating easier interpretation of the model [19].

Moreover, the process of variable selection enables models to achieve higher accuracy and better generalization capabilities. For example, van Vuuren and colleagues [20] found that LASSO created a model that was able to classify students as at risk for suicide with a higher accuracy than simple inclusion rules (i.e., predicting based on history of suicide alone). Pratik and colleagues [21] utilized Elastic Net to select variables that were able to predict smoking addiction in young adults with higher accuracy than previous research. By focusing on the most informative variables, the model can discern meaningful patterns within the data, leading to more precise predictions and improved performance on either unseen or new data. This selective approach prevents the model from being overwhelmed by noise or irrelevant information, allowing it to focus on capturing the underlying relationships that drive the outcome of interest. For example, researchers found that applying Elastic Net regularization to classifiers based on clinical notes reduced the number of features selected by more than a thousandfold, making these classifiers more easily interpretable as well as maintaining performance [22].

Furthermore, the inclusion of irrelevant variables in the modeling process can introduce bias and adversely affect the estimation of model parameters. Additional, extraneous variables may introduce noise or confounding factors, leading to skewed parameter estimates and potentially misleading conclusions [23]. By excluding such variables through proper selection techniques, we can ensure that the model's estimates remain unbiased and reflective of the true underlying relationships in the data which increases the ecological validity of study results and models produced.

Lastly, a well-selected set of variables not only enhances the model's predictive performance but also contributes to its stability and reliability [24,25]. Models built on a carefully chosen subset of variables are less susceptible to overfitting, where the model simply memorizes the data rather than learning meaningful patterns. Avoiding overfitting leads to more robust models that generalize better

and are less prone to erratic behavior or unexpected deviations which may lead to harmful classifications (e.g., classifying an individual as having a particular disorder when they do not) [26,27].

Put simply, variable selection is indispensable in the realm of machine learning. It serves as a cornerstone for improving computational efficiency, enhancing model accuracy and generalization, reducing bias in parameter estimation, and fostering the stability and reliability of the resulting models. As such, behavioral and health researchers must employ rigorous techniques and considerations during the variable selection process to ensure the effectiveness and generalizability of their models and conclusions.

The five techniques utilized in this paper were chosen for a variety of reasons. First and foremost, LASSO and Elastic Net are arguably the most popular modern variable selection techniques within the behavioral sciences. Social psychology researchers have used such techniques to create better environments that promote prosocial environments for children [28] and health researchers have used them to model the progression of Alzheimer's disease [29]. Implementations of SVM and random forest were chosen because of their strength as classification algorithms and because they can handle more complex data types (e.g., mixed variable types or non-linearly separable). Lastly, the GA was chosen 1) to introduce the reader to the concept of metaheuristic approaches to variable selection, and 2) because it has been shown to outperform more common methods like LASSO and Elastic Net across a variety of different data conditions [30].

A Motivating Example

This tutorial uses the assessment of misophonia as an example through which we illustrate each technique. Individuals with misophonia experience strong, negative, emotional responses to specific sounds (i.e., triggers) [31]. The original data sample consisted of undergraduate students (N = 343) at a large southwestern university. Participants were predominately white (76.7%), female (69.7%) and students (96.5%) ranging from ages 18 to 36 (M = 18.96, SD = 1.7). The dataset

contains 106 independent variables related to both direct characteristics of misophonia and related characteristics and one self-report binary diagnosis variable. Since misophonia is still not fully understood (i.e., diagnostic criteria have not been set, and researchers are still trying to determine the most important symptoms), this dataset functions well as an illustrative example for variable selection. It is possible that some symptoms are unimportant for, or not predictive of, a true misophonia diagnosis. One should note that this dataset does not contain any missing data, as it was handled a priori using listwise deletion. For more information on the larger previously published dataset from which this data was selected, and background on misophonia, see work by Norris and colleagues [15]. One common problem for implementing variable selection techniques is model overfitting (see Figure 1). Cross validation is one common way to circumvent overfitting. This paper implements holdout cross-validation which occurs when one splits the data into test and training sets before conducting any analyses. Typically in holdout cross-validation, 70% of the data is used for the training set and the remaining 30% is used for the test dataset. This practice was followed in this tutorial.

Methods

Logistic Regression Models

Regularization Techniques

Two of the techniques discussed in this paper, LASSO and Elastic Net, are regularization techniques. Regularization, within the context of classification problems, takes the following form:

$$L^{Reg}(\beta) = L^{logistic}(\beta) - \lambda P(\beta) \quad (1)$$

where L^{Reg} is the penalized optimization function, $L^{logistic}$ is the logistic regression function, λ is a regularization parameter (i.e., a tuning parameter), and P is a penalty function that will vary across regularization technique. The goal of regularization is to find the optimal balance between

bias (generalizability of the model) and variance (specific model fit [32]. Finding this balance is achieved via the magnitude of the lambda (λ) penalty. A larger lambda will lead to a sparser and more generalizable model. One popular technique utilized to determine the value of the lambda parameter is cross-validation. As mentioned above, cross-validation occurs when the data is split into multiple subsets, the model is developed (i.e. trained) on a subset, and evaluated (i.e., validated) on another. This process is iterative, allowing for the selection of the lambda penalty that minimizes prediction error across different subsets.

An optimal model, in the context of this paper, is one that produces the most accurate classifications. Accuracy can be calculated using the following equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

where TP is the number of individuals who were correctly classified as having a diagnosis of misophonia, TN is the number of individuals who were correctly classified as not having a diagnosis of misophonia, FP is the number of individuals who were classified as having a diagnosis but did not truly have a diagnosis in the labeled data, and FN are the number of individuals who were incorrectly classified as not having a diagnosis when a diagnosis was present in the labeled data. It is worth noting that researchers may want to use a weighted accuracy in their own research depending on the relative importance of a false positive versus a false negative. For example, a clinician attempting to predict suicide attempts may prioritize a false positive (i.e., saying the individual is likely to attempt suicide when they do not actually attempt) over a false negative (i.e., saying the individual will not attempt when they actually will). Non-weighted accuracy was chosen here for ease of explanation.

LASSO

LASSO [33] is one of two penalized regression techniques that perform variable selection. LASSO can handle data with multicollinearity, be applied to various types of data (e.g., continuous, categorical, mixed type), and is adaptable to sparse data (i.e., multiple predictors have zero or near-zero coefficients) [34,35]. The LASSO equation is as follows:

$$L^{LASSO}(\beta) = \sum_{i=1}^{P} log(P_\beta(Y_i|X_i)) - \lambda \sum_{i=1}^{P} |\beta_i| \quad (3)$$

where $L^{LASSO}(\beta)$ is the loss function, $\sum_{i=1}^{P} log(P_\beta(Y_i|X_i))$ is the summation of the MLE for all predictor variables, $\lambda \geq 0$ is the regularization hyperparameter that controls the degree of shrinkage, and β_i is a given regression coefficient. By applying the penalty (λ) to the absolute value of the beta weights, LASSO allows the coefficient associated with a given variable to be reduced (or "shrunk") to zero, eliminating those given variables from the model. For a more detailed discussion of LASSO, see Tibshirani's paper [33].

The following code utilizes the cv.glmnet() function from the glmnet package in R [36]. This function determines the magnitude of lambda through a k-fold cross-validation approach.

```
lasso.model <- cv.glmnet(x = predTrain, y = outcomeTrain, type.measure = "class", alpha=1,
                         family="binomial", nfolds = 10)
```

Through this model, we can obtain the chosen lambda value. To obtain a full list of all evaluated lambda values, use lambda.model$lambda. One can also plot the k-fold cross-validation procedure to obtain λ using plot(lasso.model) (Figure 2). Although one may use the one standard error (1se) rule to select lambda, selecting the model with prediction error one standard error above the minimum cross-validated error, this rule performs poorly in regression [37], so we use the value which minimized cross validation error (lambda min). To obtain our lambda min value, specify lasso.model$lambda.min.

Using this specified lambda value produces the coefficients seen in Table 2. Out of the original 106 predictor variables, only 16 were selected via LASSO, thus a sparse model has been obtained. The coefficient estimates obtained through a LASSO approach are biased by the nature of the algorithm [38], and thus research recommends recalculating them using a standard regression before interpreting the coefficients of the model. A comparison of the biased coefficients obtained from the LASSO model and the corrected coefficients obtained in the standard logistic model can be seen in Table 3. Obtaining the predicted classification prior to calculating accuracy is crucial. Accuracy values (Formula 2) are then determined using the coefficient estimates from both LASSO model and the logistic model. The following code can be used to obtain the accuracy values from the logistic model.

```
a.logistic <- mean(outcomeTest == pc.logistic)
```

The value obtained using the coefficient estimates from the LASSO model is an accuracy score of .86. The value obtained using the coefficient estimates from the logistic model is .89.

Despite strong performance of LASSO on this data, LASSO does have limitations [39]. First, it is unable to select more variables than there are observations. Second, LASSO will select a single variable in the presence of multicollinearity regardless of that variable's predictive capacity. To combat these first two limitations, Zou and Hastie [40] proposed a new regularization technique called Elastic Net.

Elastic Net

Elastic Net differs from LASSO through use of an additional penalty to the regression equation. Elastic Net implements both the ℓ_1 penalty, or the LASSO penalty, and the ℓ_2 penalty, or the ridge penalty, to the regression equation. With the inclusion of both penalties, the optimization function for Elastic Net is as follows:

$$L_{ElasticNet}(\beta) = \sum_{i=1}^{P} log(P_\beta(Y_i|X_i) - \lambda_1 \sum_{i=1}^{P} \beta_i^2 - \lambda_2 \sum_{i=1}^{P} |\beta_i| \quad (4)$$

where $L_{ElasticNet}(\beta)$ is the loss function, $\sum_{i=1}^{P} log(P_\beta(Y_i|X_i)$ is the summation of the MLE for all predictor variables, $\lambda_1 \geq 0$ is the regularization hyperparameter that controls the degree of shrinkage according to the ℓ_1 penalty, $\lambda_1 \geq 0$ is the regularization hyperparameter that controls the degree of shrinkage according to the ℓ_2 penalty, and β_i is a given regression coefficient. Through the inclusion of the ℓ_2 penalty, Elastic Net selects multiple variables that are highly correlated while removing all irrelevant variables [39]. Thus, if you have a dataset with highly correlated predictors (e.g., a set of dummy coded variables), Elastic Net would be more appropriate as a variable selection technique than LASSO.

We can obtain our lambda.min value using the cv.glmnet()equation as well and then use that value to fit our model using the following code.

```
en.model.min <- glmnet(x=predTrain y=outcomeTrain,
                       alpha=0.5, family="binomial", lambda =
                       elasticNet$lamda.min)
```

The only difference between this code and the code for LASSO is that alpha = 0.5 rather than 1. Coefficient estimates from the Elastic Net model and unbiased coefficients from a standard logistic model can be seen in Table 4. An accuracy of 0.88 was obtained using the coefficient estimates from the Elastic Net model while an accuracy of 0.80 was obtained using the coefficient estimates from the logistic model.

Elastic Net also has some limitations. Namely, it may struggle with datasets containing many more variables than observations, it is sensitive to outliers, and, given that it is designed for

linear relationships, it may not capture complex non-linear relationships between predictors and the response variable effectively [41].

Genetic Algorithm (GA)

Unlike LASSO and Elastic Net, which utilize internal regression models, the genetic algorithm (GA) is different. Instead of relying on a predefined internal model, the GA operates as a wrapper method. This means that the user must specify which model it should use (i.e., a user could wrap the GA around a logistic regression model or something more complex like random forest or SVM depending on the nature of their data).

A metaheuristic, such as the GA, operates at a higher level, employing strategies to efficiently explore the solution space (i.e., all possible subsets of the variables) and locate optimal or near-optimal solutions to complex optimization problems like variable selection. Unlike specific algorithms tailored to certain types of problems, metaheuristics like the GA offer flexibility and adaptability, making them suitable for a wide range of applications.

The GA, inspired by the principles of natural selection and evolution, mimics the process of biological evolution to iteratively refine potential solutions. Through mechanisms such as crossover, mutation, and selection, the GA explores and evolves a population of potential solutions over successive generations, gradually improving the overall quality of solutions. Figure 3 illustrates the general structure of the genetic algorithm, depicting its iterative process of generating, evaluating, and evolving solutions. Each iteration refines the population, guiding the search towards promising regions of the solution space.

For a comprehensive understanding of the genetic algorithm and its application to variable selection, interested readers are encouraged to refer to the work by Bain and colleagues [30]. Their research provides detailed insights into the underlying principles, implementation strategies, and

practical considerations associated with the GA's use in solving two-group classification optimization problems.

For this paper, logistic regression is chosen as the model around which the GA will wrap. The optimization function used in this paper is the Hubert and Arabie [42] Adjusted Rand Index (ARI). ARI is a measure of agreeability between predicted classifications and true (or known) classifications and can be calculated in the following way:

$$ARI = \frac{RI - RI_{Expected}}{max(RI) - RI_{Expected}} \quad (5)$$

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

The R code needed to implement both logistic regression and ARI can be found on lines 248 through 260 in the companion code file on OSF. For a guide of hyperparameters and their default values (if a default is selected), see Table 5. To implement the GA, the following code can be run:

```
ga.solution <- ga(fitness = function(vars)
  gaOpt(vars=vars, IV.train=data.frame(predTrain),
    DV.train=outcomeTrain),
    type = "binary", nBits = ncol(predTrain),
    names = colnames(predTrain), seed = 123456,
    run=5
)
```

To view the selected subset of variables from the ga() function, one calls, ga.solution@solution[1,]. Note, the returned solution (given by ga.solution@solution) contains many potential subsets of variables, but by referencing only the first row (using the indexing [1,]), the optimal subset of variables as determined by the GA can be accessed. Since the ga() function does not have a specified method for model building, but rather simply returns a list of variable selections, one must first build a model to obtain an accuracy value for the selected variables.

Given that the internal model we specified was a logistic regression model, it makes sense to use a simple logistic model which can be built using the following code. The coefficients from this model can be seen in Table 6.

```
allVarNames <- colnames(predTrain)
selectedVarNames <- allVarNames[ga.solution@solution[1,]==1]
selectedVars <- data.frame(predTest[,selectedVarNames],
   outcomeTest)
ga.model <- glm(outcomeTest~., family='binomial',
   data=selectedVars)
```

After building the model, an accuracy can be obtained using the following code:

```
p <- predict(ga.model, newx = predTest)
c <- ifelse(p >= .8, 1,0)
accuracy <- mean(c == outcomeTest)
```

An accuracy value of 1 is obtained, indicating overfitting, as with the past models built in this tutorial. Current literature indicates that the GA is prone to overfitting [43–45], suggesting the model would not fit quite as well if a new sample was collected, despite the accuracy of the model fit for the test sample used in this tutorial.

Support Vector Machines

Support Vector Machines (SVM) are a class of supervised learning models widely employed in classification and regression tasks [46,47]. SVMs operate by finding the optimal hyperplane that maximizes the margin between different classes of data points. By maximizing the margin between classes, SVM achieves good generalizability and is robust to outliers [48,49]. SVM can handle both linearly separable and non-linearly separable data by use of a kernel function which artificially projects the original data into a higher-dimensional space [47].

Elastic SCAD SVM

SVM by itself is a classification algorithm. However, researchers have created implementations of SVM that simultaneously perform classification and variable selection [50–52].

This tutorial uses an approach like LASSO and Elastic Net in that it selects variables via the addition of a penalty [50]. The penalty utilized in this tutorial is the Elastic SCAD penalty which when included in an SVM, reads:

$$SVM_{ElasticSCAD} = min_{b,w}(sign(\boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{b}) + \sum_{j=1}^{p} P_{SCAD}\,\lambda_1(W_j) + \lambda_2\|w\|_2^2) \quad (6)$$

where $\lambda_1,\lambda_2 \geq 0$ control the degree of shrinkage applied by the SCAD ($P_{SCAD}\lambda_1(W_j)$) and Elastic Net ($\lambda_2||w||_2^2$) penalties, respectively. For more information on the SCAD penalty, see work by Becker and colleagues [53]. The initial part of the equation ($sign(\boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{b})$) is the base equation for an SVM where $\boldsymbol{w}$ is the weight vector, $\boldsymbol{x}$ is the input feature vector, $\boldsymbol{b}$ is the bias term vector, $sign(.)$ is the sign function, which returns +1 if the argument is positive, -1 if negative, and 0 if zero. All hyperparameters are set to default values in this tutorial. In addition, data needs to be restructured for this function. For a clearer understanding of the additional hyperparameters in the svmfs() function, see Table 7. The svmfs() function can be applied in the following manner.

```
Bounds <- t(data.frame(log2lambda1=c(-10, 10), log2lambda2=c(-
        10,10)))
colnames(bounds)<-c("lower", "upper")
svm.model <- svmfs(x=predTrain, y = svmTrainOutcome, fs.method =
        "scad+L2", bounds=bounds, grid.search = "interval",        inner.val.method =
"cv", show = "none", parms.coding = "none",
        seed=123456)
```

The output of the model created using the svmfs() function has its own nomenclature that requires explanation. First, rather than referring to the coefficients as coefficients, the model uses the w parameter (coming from the term beta weight). The b parameter illustrates the intercept of the SVM hyperplane and can be thought of like the b_0 of a regression model. The xind parameter tells the user the index (or column location) of the variables selected in the dataset. The full

output can be seen in Table 8. To examine the accuracy of this model, the same predict function can be used as was implemented previously, but the outputted predictions will require some restructuring, as they come in the form of a factor with underlying numeric values 1 and 2 and they need to have numeric values of 0 and 1. The Elastic SCAD SVM model obtained an accuracy of 0.83. The code required to calculate that accuracy value is below.

```
esvm.predictions <- predict(svm.model, newdata = svmTestPreds) esvm.predictions.formatted <-
as.numeric(esvm.predictions$pred.class)-1 esvm.accuracy <- mean(esvm.predictions.formatted
== outcomeTest)
```

Limitations of SVM include the researcher's selection of the kernel function, computation time, and dimension constraints. By default, the svmfs() function utilizes a linear kernel function. Since the kernel is chosen a priori by the researcher, for optimal results, an optimal function must be used. SVM models are computationally more expensive than a simpler classification technique (e.g., logistic regression) and will take longer to compute. SVM models face the same degrees of freedom problem as LASSO and Elastic Net, where it is limited by the number of observations. As such, an ideal dataset for SVM would contain more observations than variables.

Tree Based Models

Random Forest

Another powerful classifier is a decision (or classification) tree. An example can be seen in Figure 4. From this decision tree, it can be concluded that anyone whose score on variable S5_57 is less than 3 and score on variable S5_60 is less than 3 does not qualify for a misophonia diagnosis. Decision trees are not only powerful classifiers, but they also produce an output that is easy to interpret. However, decision trees are prone to overfitting – so much so that overfitting is almost guaranteed [54]. One of the most efficient ways to avoid overfitting is by using multiple trees (i.e., creating a random forest). Random forest creates many decision trees using a randomly selected subset of the data to create each individual tree. The results of all trees are then

aggregated to predict the desired outcome. Some major benefits of a random forest classifier are that it can be used with an outcome variable that has any number of levels [55], meaning that unlike logistic regression which only works with binary variables, random forest could handle a variable with 3, 4, or even 10 different levels. However, these trees are only used for classification, meaning that they do not perform variable selection. Thus, researchers have had to adapt the classifier to perform variable selection. The utilization of random forest in the Boruta package performs well in many different conditions [56], and therefore, is the implementation demonstrated in this tutorial.

The Boruta package contains a series of functions pertaining to variable selection techniques using different measures of importance to select the variables. A measure of importance simply indicates the value of a given variable to the overall strength of the model. The variables that are more useful, meaning that they are stronger predictors of the outcome variable, are deemed more important, and thus are more likely to be selected than those of a lesser importance (i.e., less predictive power). To run the model, a simple regression formula statement is used: outcome ~ predictors. Because all predictors will be used, a shortcut can be implemented through use of a period (.) in place of predictors as is seen in the code below. If not, all variables were to be included in the model, the user would need to type all the relevant predictors names in the formula statement concatenated with addition symbols (+). Knowing this, the model can then be built using the following code:

```
set.seed(123456)
boruta.model <- Boruta(as.factor(MQDX) ~. , data=trainDat)
```

The Boruta() function classifies variables as either important, unimportant, or of tentative importance. Regarding the misophonia dataset, 15 variables were deemed important, 74 variables were deemed unimportant, and the remaining 17 variables were placed in the tentative category. For a list of all variables that were classified in each category and a visualization of the

boruta.model output, see Table 9. Figure 5 illustrates variability of the importance score calculated for each variable during the Boruta process, as well as their ultimate classification. A model can be built using either a) all variables that were not deemed unimportant (non-rejected variables) or b) only the confirmed important variables. For the purpose of this tutorial, only variables which have been confirmed important are included in the model. This model is then built using the randomForest() function since Boruta() internally implements a random forest model. The model is built in the following way.

```
set.seed(123456)
selectedModel <- randomForest(getConfirmedFormula(boruta.model),
   data=trainDat)
```

The predictive accuracy of the random forest model can be calculated using the predict() function, just as it has been for other models. An accuracy of .88 was obtained for this model. The algorithm may not perform well with highly unbalanced classifications or in situations where a given level contains a very small number of classifications.

Comparing All Models

For a comparison of the accuracy values obtained by all techniques implemented in this tutorial, see Table 10. From this, we can state that the GA produced the most accurate model. However, there was no difference in the LASSO non-biased, Boruta, and Elastic Net models in terms of accuracy. Depending on the purpose of your model, you may want to use a performance metric other than accuracy. Within the context of our motivating example, it may be worth examining the following:

- Sensitivity: given the individual truly has misophonia, how likely is the classifier to realize that?

- Specificity: given the individual truly does not have misophonia, how likely is the classifier to realize that?
- Positive predictive value: given the classifier claims the individual to have misophonia, how likely is it that the individual really has misophonia?
- Negative predictive value: given the classifier claims the individual does not have misophonia, how likely is it that the individual really does not have the disease?

Given that our example pertains to diagnosis, it is possible that one may favor sensitivity over specificity in that we want to minimize the number of missed cases. However, it is also possible that we would want to minimize the number of false diagnoses to save individuals the cost of unnecessary intervention. A confusion matrix (discussed briefly in Appendix B) might be useful. Alternatively, one could use the Area Under the Curve (AUC) of the Receiver operating characteristic (ROC) curve. One should carefully consider these factors when deciding on the performance metric by which to evaluate a model.

We can also examine the different variables each method selected (Tables 2 – 4, 6, and 8 – 9). The GA selected the most variables which likely accounts for its very strong accuracy. However, Elastic SCAD SVM selected many more variables than LASSO, but had a worse accuracy. Given this outcome, it may not be ideal to use all variables selected by Elastic SCAD SVM in this dataset. There was only one variable (S5_57) that was selected by all five methods. So, there is a clear method effect on the variables that are deemed to be important. Within the context of our example, we could interpret this to mean that the question, “Sometimes in response to sounds I feel rage that is difficult to control” is an incredibly important predictor for misophonia and may capture a defining characteristic of the disorder.

Discussion

This tutorial provided an overview and a practical guide for implementation of LASSO [36], Elastic Net [36], a genetic algorithm [57,58], Elastic SCAD SVM [50], and random forest via Boruta [56] in R v. 4.2.1. Proper analysis of the output as well as comparisons on the predictive accuracy of each method are also discussed. More information on R, other useful machine learning software, and some of these functions were provided in the Appendices. Lastly, an OSF project containing all code implemented in this tutorial, as well as the data used, is available.

Variable selection allows researchers to find parsimonious models that are also good predictive or classifying models. Given R's increasing popularity among researchers due to the free and open access nature of the software, it is of value to the field to provide more guidance on the variable selection methods available in R. In addition, the extent to which some of these methods overfit data should not be ignored when implementing them on real-world data. If a researcher is concerned with creating a generalizable model, it is recommended to validate the results not only through some form of cross-validation, but also through collection of a new sample. Through this tutorial, we aim to push the field towards more transparent guidelines and standardization for the use of variable selection techniques and machine learning in psychological research.

There are many additional R packages that will perform variable selection using random forest as well as SVMs, but only one of each was demonstrated in this tutorial. The demonstrated methods in the current tutorial were selected because they are commonly used in the psychological sciences, are powerful techniques for both classification (e.g., diagnosing individuals with misophonia) and variable selection, and because they are all freely available to researchers in R. In a similar vein, we have included only five machine learning methods here but many more exist, and additional tutorials should be provided to applied researchers about how

best to implement them following research demonstrating each algorithm's performance to indicate which algorithm is best for addressing certain research questions. For the interested reader, a comparison of the performance of each method demonstrated in this tutorial can be found in Bain and colleagues [30].

This tutorial presented instructions for using five variable selection techniques in R which serves as a useful starting point for understanding how to perform variable selection. We encourage the user to refer to each method's full documentation for additional examples and detail. We hope that this tutorial makes these methods more easily accessible to the everyday psychological researcher, opens doors to applications of variable selection in new areas, and leads to a decreased presence of less powerful methods (e.g., stepwise selection) in the literature.

Data Availability

The accompanying code and data utilized in this tutorial can be found here: https://osf.io/pr6j8/?view_only=de4405aae6fd4bc7bffb85f0e872216. Additional supplementary information such as a glossary of key terms, R package recommendations, etc. are also available through OSF.

References

1. Lenters, V., Vermeulen, R., & Portengen, L. (2018). Performance of variable selection methods for assessing the health effects of correlated exposures in case–control studies. *Occupational and Environmental Medicine*, *75*(7), 522–529. https://doi.org/10.1136/oemed-2016-104231

2. Amene, E., Hanson, L. A., Zahn, E. A., Wild, S. R., & Döpfer, D. (2016). Variable selection and regression analysis for the prediction of mortality rates associated with foodborne diseases. *Epidemiology and Infection*, *144*(9), 1959–1973. https://doi.org/10.1017/S0950268815003234

3. Bourdès, V., Bonnevay, S., Lisboa, P., Defrance, R., Pérol, D., Chabaud, S., Bachelot, T., Gargi, T., & Négrier, S. (2010). Comparison of Artificial Neural Network with Logistic Regression as Classification Models for Variable Selection for Prediction of Breast Cancer Patient Outcomes. *Advances in Artificial Neural Systems*, *2010*, 1–11. https://doi.org/10.1155/2010/309841

4. Wettstein, A., Jenni, G., Schneider, I., Kühne, F., grosse Holtforth, M., & La Marca, R. (2023). Predictors of Psychological Strain and Allostatic Load in Teachers: Examining the Long-Term Effects of Biopsychosocial Risk and Protective Factors Using a LASSO Regression Approach. *International Journal of Environmental Research and Public Health*, *20*(10), Article 10. https://doi.org/10.3390/ijerph20105760

5. Luo, J., Ren, S., Li, Y., & Liu, T. (2021). The Effect of College Students' Adaptability on Nomophobia: Based on Lasso Regression. *Frontiers in Psychiatry*, *12*. https://www.frontiersin.org/articles/10.3389/fpsyt.2021.641417

6. Derksen, S., & Keselman, H. J. (1992). Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of*

Mathematical and Statistical Psychology, *45*(2), 265–282. https://doi.org/10.1111/J.2044-8317.1992.TB00992.X

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

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

9. Wiegand, R. E. (2010). Performance of using multiple stepwise algorithms for variable selection. *Statistics in Medicine*, *29*(15), 1647–1659. https://doi.org/10.1002/sim.3943

10. Thompson, B. (1995). Stepwise Regression and Stepwise Discriminant Analysis Need Not Apply here: A Guidelines Editorial. *Educational and Psychological Measurement*, *55*(4), 525–534. https://doi.org/10.1177/0013164495055004001

11. Smith, G. (2018). Step away from stepwise. *Journal of Big Data*, *5*(1), 32. https://doi.org/10.1186/s40537-018-0143-6

12. Serang, S., Jacobucci, R., Brimhall, K. C., & Grimm, K. J. (2017). Exploratory Mediation Analysis via Regularization. *Structural Equation Modeling : A Multidisciplinary Journal*, *24*(5), 733–744. https://doi.org/10.1080/10705511.2017.1311775

13. Shi, D., Shi, D., & Fairchild, A. J. (2023). Variable Selection for Mediators under a Bayesian Mediation Model. *Structural Equation Modeling: A Multidisciplinary Journal*, *0*(0), 1–14. https://doi.org/10.1080/10705511.2022.2164285

14. Gunn, H. J., Hayati Rezvan, P., Fernández, M. I., & Comulada, W. S. (2023). How to apply variable selection machine learning algorithms with multiply imputed data: A missing discussion. *Psychological Methods*, *28*(2), 452–471. https://doi.org/10.1037/met0000478

15. Norris, J. E., Kimball, S. H., Nemri, D. C., & Ethridge, L. E. (2022). Toward a Multidimensional Understanding of Misophonia Using Cluster-Based Phenotyping. *Frontiers in Neuroscience*, *16*. https://doi.org/10.3389/fnins.2022.832516

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

17. Chowdhury, M. Z. I., & Turin, T. C. (2020). Variable selection strategies and its importance in clinical prediction modelling. *Family Medicine and Community Health*, *8*(1), e000262. https://doi.org/10.1136/fmch-2019-000262

18. Barceló, P., Monet, M., Pérez, J., & Subercaseaux, B. (2020). Model interpretability through the lens of computational complexity. *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 15487–15498.

19. Jia, W., Sun, M., Lian, J., & Hou, S. (2022). Feature dimensionality reduction: A review. *Complex & Intelligent Systems*, *8*(3), 2663–2693. https://doi.org/10.1007/s40747-021-00637-x

20. van Vuuren, C. L., van Mens, K., de Beurs, D., Lokkerbol, J., van der Wal, M. F., Cuijpers, P., & Chinapaw, M. J. M. (2021). Comparing machine learning to a rule-based approach for predicting suicidal behavior among adolescents: Results from a longitudinal population-based survey. *Journal of Affective Disorders*, *295*, 1415–1420. https://doi.org/10.1016/j.jad.2021.09.018

21. Pratik, S., Nayak, D., Prasath, R. R., & Swarnkar, T. (2022). *Prediction of Smoking Addiction Among Youths Using Elastic Net and KNN: A Machine Learning Approach* (pp. 199–209). https://doi.org/10.1007/978-3-031-21517-9_20

22. Marafino, B. J., John Boscardin, W., & Adams Dudley, R. (2015). Efficient and sparse feature selection for biomedical text classification via the elastic net: Application to ICU risk

stratification from nursing notes. *Journal of Biomedical Informatics*, *54*, 114–120. https://doi.org/10.1016/j.jbi.2015.02.003

23. Kerkhoff, D., & Nussbeck, F. W. (2019). The Influence of Sample Size on Parameter Estimates in Three-Level Random-Effects Models. *Frontiers in Psychology*, *10*. https://doi.org/10.3389/fpsyg.2019.01067

24. Arjomandi-Nezhad, A., Guo, Y., Pal, B. C., & Varagnolo, D. (2023). *A Model Predictive Approach for Enhancing Transient Stability of Grid-Forming Converters* (arXiv:2308.01020). arXiv. http://arxiv.org/abs/2308.01020

25. Fox, E. W., Hill, R. A., Leibowitz, S. G., Olsen, A. R., Thornbrugh, D. J., & Weber, M. H. (2017). Assessing the accuracy and stability of variable selection methods for random forest modeling in ecology. *Environmental Monitoring and Assessment*, *189*(7), 316. https://doi.org/10.1007/s10661-017-6025-0

26. Cateni, S., Colla, V., & Vannucci, M. (2010). Variable Selection through Genetic Algorithms for Classification Purposes. *Artificial Intelligence and Applications*. Artificial Intelligence and Applications, Innsbruck, Austria. https://doi.org/10.2316/P.2010.674-080

27. Heinze, G., Wallisch, C., & Dunkler, D. (2018). Variable selection – A review and recommendations for the practicing statistician. *Biometrical Journal. Biometrische Zeitschrift*, *60*(3), 431–449. https://doi.org/10.1002/bimj.201700067

28. Chu, M., Fang, Z., Mao, L., Ma, H., Lee, C.-Y., & Chiang, Y.-C. (2024). Creating A child-friendly social environment for fewer conduct problems and more prosocial behaviors among children: A LASSO regression approach. *Acta Psychologica*, *244*, 104200. https://doi.org/10.1016/j.actpsy.2024.104200

29. Liu, X., Cao, P., Gonçalves, A. R., Zhao, D., & Banerjee, A. (2018). Modeling Alzheimer's Disease Progression with Fused Laplacian Sparse Group Lasso. *ACM Transactions on Knowledge Discovery from Data*, *12*(6), 65:1-65:35. https://doi.org/10.1145/3230668

30. Bain, C., Shi, D., Boness, C. L., & Loeffelman, J. (2023). *A Simulation Study Comparing the Use of Supervised Machine Learning Variable Selection Methods in the Psychological Sciences*. PsyArXiv. https://doi.org/10.31234/osf.io/y53t6

31. Wu, M. S., Lewin, A. B., Murphy, T. K., & Storch, E. A. (2014). Misophonia: Incidence, Phenomenology, and Clinical Correlates in an Undergraduate Student Sample: Misophonia. *Journal of Clinical Psychology*, *70*(10), 994–1007. https://doi.org/10.1002/jclp.22098

32. Helwig, N. E. (2017). Adding bias to reduce variance in psychological results: A tutorial on penalized regression. *The Quantitative Methods for Psychology*, *13*(1), 1–19. https://doi.org/10.20982/tqmp.13.1.p001

33. Tibshirani, R. (1996). *Bias, variance and prediction error for classification rules*. University of Toronto.

34. Foucart, S., Tadmor, E., & Zhong, M. (2023). On the Sparsity of LASSO Minimizers in Sparse Data Recovery. *Constructive Approximation*, *57*(2), 901–919. https://doi.org/10.1007/s00365-022-09594-1

35. Mendez-Civieta, A., Aguilera-Morillo, M. C., & Lillo, R. E. (2021). Adaptive sparse group LASSO in quantile regression. *Advances in Data Analysis and Classification*, *15*(3), 547–573. https://doi.org/10.1007/s11634-020-00413-8

36. Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, *33*(1). https://doi.org/10.18637/jss.v033.i01

37. Chen, Y., & Yang, Y. (2021). The One Standard Error Rule for Model Selection: Does It Work? *Stats*, *4*(4), 868–892. https://doi.org/10.3390/stats4040051

38. Yarkoni, T., & Westfall, J. (2017). Choosing Prediction Over Explanation in Psychology: Lessons From Machine Learning. *Perspectives on Psychological Science*, *12*(6), 1100–1122. https://doi.org/10.1177/1745691617693393

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

40. Zou, H., & Hastie, T. (2005). Regularization and Variable Selection via the Elastic Net. *Journal of the Royal Statistical Society.*, *67*(2), 301–320.

41. Wang, L., Cheng, H., Liu, Z., & Zhu, C. (2014). A robust elastic net approach for feature learning. *Journal of Visual Communication and Image Representation*, *25*(2), 313–321. https://doi.org/10.1016/j.jvcir.2013.11.002

42. Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*(1), 193–218. https://doi.org/10.1007/BF01908075

43. Cunningham, P., & Loughrey, J. (2005). Overfitting in Wrapper-Based Feature Subset Selection: The Harder You Try the Worse it Gets. *Research and Development in Intelligent Systems XXI*, 33–43. https://doi.org/10.1007/1-84628-102-4_3

44. Fröhlich, H., Chapelle, O., & Schölkopf, B. (2003). *Feature Selection for Support Vector Machines by Means of Genetic Algorithms*. https://doi.org/10.1109/TAI.2003.1250182

45. Leardi, R. (2000). Application of genetic algorithm-PLS for feature selection in spectral data sets. *Journal of Chemometrics*, *14*, 643–655.

46. Fernandez, M., Caballero, J., Fernandez, L., & Sarai, A. (2011). Genetic algorithm optimization in drug design QSAR: Bayesian-regularized genetic neural networks (BRGNN) and genetic

algorithm-optimized support vectors machines (GA-SVM). *Molecular Diversity*, *15*(1), 269–289. https://doi.org/10.1007/s11030-010-9234-9

47. Karatzoglou, A., Meyer, D., & Hornik, K. (2006). Support Vector Machines in *R*. *Journal of Statistical Software*, *15*(9). https://doi.org/10.18637/jss.v015.i09

48. Singla, M., & Shukla, K. K. (2020). Robust statistics-based support vector machine and its variants: A survey. *Neural Computing and Applications*, *32*(15), 11173–11194. https://doi.org/10.1007/s00521-019-04627-6

49. Xu, H., Caramanis, C., & Mannor, S. (2009). Robustness and Regularization of Support Vector Machines. *Journal of Machine Learning Research 1*, *10*, 1485–1510.

50. Becker, N., Werft, W., & Benner, A. (2018). *penalizedSVM: Feature Selection SVM using Penalty Functions* [Computer software]. https://CRAN.R-project.org/package=penalizedSVM

51. Bierman, S., & Steel, S. (2009). Variable selection for support vector machines. *Communications in Statistics: Simulation and Computation*, *38*(8), 1640–1658. https://doi.org/10.1080/03610910903072391

52. Tharwat, A., & Hassanien, A. E. (2019). Quantum-Behaved Particle Swarm Optimization for Parameter Optimization of Support Vector Machine. *Journal of Classification*, *36*, 576–598. https://doi.org/10.1007/s00357-018-9299-1

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

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

55. Brieuc, M. S. O., Waters, C. D., Drinan, D. P., & Naish, K. A. (2018). A practical introduction to Random Forest for genetic association studies in ecology and evolution. *Molecular Ecology Resources*, *18*(4), 755–766. https://doi.org/10.1111/1755-0998.12773

56. Kursa, M. B., & Rudnicki, W. R. (2010). Feature Selection with the **Boruta** Package. *Journal of Statistical Software*, *36*(11). https://doi.org/10.18637/jss.v036.i11

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

58. Scrucca, L. (2017). On some extensions to GA package: Hybrid optimisation, parallelisation and islands evolution. *R Journal*, *9*(1), 187–206. https://doi.org/10.32614/rj-2017-008

59. Ghojogh, B., & Crowley, M. (2019). The Theory Behind Overfitting, Cross Validation, Regularization, Bagging, and Boosting: Tutorial. In *arXiv*. http://arxiv.org/abs/1905.12787

Additional Information

List of author contributions: CMB conducted all analyses and drafted the manuscript; regarding the data utilized here, JEN aided in the original study design, led all data collection and data preprocessing while LEE conceived of the study design of the project and supervised all aspects of funding, participant recruitment, and data collection; JEL supervised data analysis and manuscript preparation. DS supervised manuscript preparation and provided guidance about organization and focus of the paper. All authors contributed significantly to manuscript preparation.

No authors have any competing interests to report at this time.

Catherine M. Bain. https://orcid.org/0000-0002-2767-6882

Dingjing Shi. https://orcid.org/0000-0002-5652-3818

Lauren E. Ethridge. https://orcid.org/0000-0003-0601-6911

Jordan E. Norris. https://orcid.org/0000-0002-4438-3416

Jordan E. Loeffelman. https://orcid.org/0000-0002-0269-7708

Appendix A

Demographic information on the original sample:

Variable	
Age (Years)	*M* = 18.96 *SD* = 1.7
Gender	
Male	104 (30.3%)
Female	239 (69.7%)
Ethnicity	
White	263 (76.7%)
Black/African American	32 (9.3%)
Latino/Hispanic	46 (13.4%)
Asian/Asian American	28 (8.2%)
American Indian/Alaska Native	26 (7.6%)
Native Hawaiian/Other Pacific Islander	2 (0.6%)
Other	2 (0.6%)
Education	
Less than high school	2 (0.6%)
High school graduate	129 (37.6%)
Some years of college/university (no degree)	194 (56.6%)
Vocational training	2 (0.6%)
Associates degree	8 (2.3%)
Bachelor’s degree	5 (1.5%)
Master’s degree	1 (0.3%)

Appendix B

The output of a randomForest() model takes the following form:

```
##
## Call:
## randomForest(formula = getConfirmedFormula(boruta.model), data = trainDat)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3 ##
##         OOB estimate of error rate: 0%
## Confusion matrix:
##       0 1 class.error
## 0 124 0           0
## 1   0 24          0
```

The random forest output contains different information than any other technique discussed in this paper because it performs a type of cross-validation internally through looking at something called Out of Bag error (OOB; sometimes referred to as the out-of-bag estimate). The OOB is an approach to measuring the prediction error of a random forest model or of other decision tree models. OOB error is the mean prediction error of a given sample, using only the trees which did not have that sample in their bootstrapped sample. This sounds potentially confusing, but it simply means that the OOB error is the average prediction error of a given sample of data when that sample of data is treated as a test sample rather than a train sample (i.e., a tree is evaluated on that data since it has yet to see it). OOB error is also used for other machine learning models implementing something called bootstrap aggregation (bagging). Bagging is the

official term for only considering a random sample of the data when random forest creates each tree. It is unique in that it is a random sample that allows for repetition, meaning that the records for a single participant could be represented more than once in the sample. For more on the theory behind bagging, see work by Ghojogh and Crowley [59].

In addition to the OOB error rate, the output provides a confusion matrix, something that is often used to discuss the performance of a classification method. A confusion matrix follows the form below:

	True 0	True 1
Predicted 0	Correct Rejection	Miss
Predicted 1	False Alarm	Hit

It is ideal to have a high number of both hits and correct rejections and a low number of both false alarms and misses. It is possible that one may wish to allow for more false alarms so as to decrease miss rates in some cases (e.g., a doctor would likely rather have a false positive screening for cancer than miss a cancer diagnosis). In other cases, one may want to minimize false alarms (e.g., in the court system, it is ideal to minimize the number of innocent people who are sent to jail). Thus, it is incredibly beneficial to understand each of these statistics when evaluating the performance of a classification model, as they both factor into calculating accuracy.

The randomForest() output provides a classification error representing the proportion of a given class which has been misclassified (e.g., a true 0 that was classified as 1 or the reverse). For the model demonstrated, there is no classification error for either class since perfect accuracy occurred.

Figures

Table 1

A table containing the hyperparameters of the cv.glmnet() function and their corresponding definitions.

Parameter	Description
x	A matrix of predictor (or input) variables
y	The vector containing the response (or outcome) variable
type.measure	The optimization measure to be used within the internal cross-validation procedure. By setting this to "class" misclassification error is optimized.
alpha	The Elastic Net mixing hyperparameter. Because the same function is used to implement ridge, LASSO, and Elastic Net, the value for alpha determines which regularization technique is run. Alpha is constrained between 0 and 1, with a value of 0 implementing ridge regression, 1 implementing LASSO regression, and anything in between implementing an Elastic Net regression.
family	The type of regression to be implemented. By setting this hyperparameter to "binomial" an MLE regression is implemented.
nfolds	The number of partitions implemented in the internal k-fold cross-validation.

Table 2

A table containing the variables selected by the LASSO model and their corresponding estimated coefficients.

Variable	Coefficient
(Intercept)	-5.741
MQ4	0.154
MQ11	0.039
MQ12	0.137
MQ13	0.112
MQ17	0.045
S5_7	0.087
S5_24	-0.032
S5_25	0.318
S5_31	0.159
S5_32	0.024
S5_35	0.123
S5_36	0.089
S5_56	0.124
S5_57	0.419
S5_75	0.252
S5_78	-0.018

Table 3

A table containing the variables selected by the LASSO model and the coefficient estimates obtained directly from the LASSO model as well as the re-estimated (non-biased) coefficients obtained by creating a typical logistic model using the selected variables.

Variable	LASSO Estimate	Logistic Estimate
(Intercept)	-5.741	-8.802
MQ4	0.154	0.361
MQ11	0.039	0.480
MQ12	0.137	0.760
MQ13	0.112	-0.193
MQ17	0.045	0.143
S5_7	0.087	-0.057
S5_24	-0.032	-1.154
S5_25	0.318	1.051
S5_31	0.159	0.538
S5_32	0.024	0.245
S5_35	0.123	0.110
S5_36	0.089	0.285
S5_56	0.124	0.387
S5_57	0.419	0.867
S5_75	0.252	0.186
S5_78	-0.018	-0.600

Table 4

A table containing the variables selected by the Elastic Net model and their corresponding estimated coefficients obtained directly from the Elastic Net model as well as the coefficients estimated by implementing a logistic model (non-biased coefficients).

Variable	Elastic Net Estimate	Logistic Estimate
(Intercept)	-5.796	-1732.902
MO4	0.133	15.606
MO11	0.046	20.788
MO12	0.119	81.502
MO13	0.103	-6.765
MO15	0.057	101.386
MO16	0.075	5.368
MO17	0.086	145.615
S5_2	0.023	-6.618
S5_7	0.091	12.560
S5_11	-0.051	-190.866
S5_24	-0.095	-39.021
S5_25	0.262	31.171
S5_27	0.031	94.479
S5_31	0.165	58.559
S5_32	0.092	97.889
S5_35	0.147	56.132
S5_36	0.088	38.234
S5_38	0.036	30.691
S5_40	0.048	111.532
S5_42	0.032	42.788
S5_53	-0.020	12.132
S5_56	0.19	94.590
S5_57	0.259	-87.586
S5_68	0.081	126.088
S5_74	0.018	5.356
S5_75	0.197	-17.256
S5_78	-0.089	-101.315
S5_82	0.033	-2.060

Table 5

A table containing the hyperparameters of the ga() function and their corresponding definitions and default values.

Parameter	Description
fitness	The hyperparameter containing the optimization function is passed. No default is set.
type	The type of ga that needs to be run is dependent upon the nature of the outcome variable. 'binary' is selected.
crossover	The type of crossover performed. The default for a binary implementation is found via the 'ga_Crossover()' function.
popSize	An R function to generate the initial population. To access available functions, run 'ga_Population()'.
pcrossover	The probability of crossover, default of 0.8 is used.
pmutation	The probability of mutation, default of 0.1 is used.
elitism	The number of best fitted chromosomes to survive at the end of each generation, default of max(1, round(popSize*0.05)) is used.
nBits	A value specifying the number of bits in a potential solution, set equal to the number of predictors.
names	The variable names.
maxIter	The maximum number of iterations to run before the GA search is halted, default of 100 is used.
keepBest	A logical argument specifying if best solutions at each iteration should be saved, default FALSE.
seed	A number allowed to control randomness for reproducibility.
run	The number of consecutive generations that can occur without any improvement before the GA is halted, default is modified from maxiter to 5.

Table 6

A table containing the variables selected by the GA and their corresponding estimated coefficients in the logistic regression model.

Variable	Coefficient	Variable	Coefficient
(Intercept)	72.896	S5_42	-8.713
MQ4	-7.952	S5_45	8.668
MQ6	-3.454	S5_46	-10.066
MQ8	-5.707	S5_49	-1.448
MQ17	-6.154	S5_50	5.708
MQ18	21.166	S5_51	-0.198
S5_2	9.767	S5_52	-8.592
S5_3	-3.703	S5_53	-2.791
S5_4	-0.814	S5_55	-13.721
S5_6	3.907	S5_57	3.470
S5_7	-18.858	S5_58	-2.086
S5_8	1.915	S5_60	-16.660
S5_9	14.258	S5_62	4.408
S5_10	10.305	S5_63	5.143
S5_11	-11.589	S5_64	-0.372
S5_12	43.946	S5_65	4.143
S5_13	-36.764	S5_66	-8.781
S5_18	10.628	S5_68	-13.752
S5_19	2.410	S5_69	7.001
S5_20	-6.446	S5_72	12.343
S5_21	-0.442	S5_73	19.055
S5_23	3.657	S5_76	-9.715
S5_25	2.824	S5_77	-4.178
S5_26	1.490	S5_78	5.347
S5_27	4.120	S5_79	-7.315
S5_31	-4.880	S5_81	7.567
S5_32	-14.408	S5_83	-4.304
S5_33	-11.206	S5_84	-1.235
S5_38	5.880	S5_86	5.970
S5_41	11.462	S5_87	-14.504

Table 7

A table containing the hyperparameters of the svmfs() function as well as their corresponding definitions.

Parameter	Description
x	Matrix of the input or predictor variables where the columns are the variables, and the rows are the observations.
y	A numerical vector of class labels, -1, 1.
fs.method	The feature (or variable) selection method. Available methods include 'scad', '1norm' used for LASSO, 'DrHSVM' for Elastic Net, and 'scad+L2; for Elastic SCAD.
bounds	For an interval grid search a list of values for lambda1 and lambda 2 must be provided to the model.
grid.search	The inner validation method used to obtain the values for lambda1 and lambda2.
inner.val.method	Whether or not the plots of DIRECT algorithm should be shown.
show	Specification of how hyperparameters should be recoded or if no recoding should occur.
parms.coding	By specifying a seed, the results become reproducible. It is included here for the sake of those readers following along.
seed	Matrix of the input or predictor variables where the columns are the variables, and the rows are the observations.

Table 8

A table containing the full output of the SVM model.

Variable	Coefficient	Variable	Coefficient
(Intercept)	-1.209	S5_38	0.003
MQ3	0.002	S5_39	0.003
MQ5	0.003	S5_40	0.001
MQ8	0.003	S5_41	-0.002
MQ11	0.002	S5_42	0.002
MQ12	0.003	S5_43	0.002
MQ16	0.003	S5_49	0.002
MQ17	0.003	S5_53	-0.003
MQ18	0.002	S5_55	0.003
S5_1	0.001	S5_56	0.007
S5_2	0.005	S5_57	0.005
S5_7	0.006	S5_59	0.003
S5_10	-0.003	S5_65	-0.005
S5_11	-0.002	S5_66	0.001
S5_13	-0.001	S5_68	0.004
S5_24	-0.005	S5_69	0.001
S5_25	0.006	S5_72	0.001
S5_26	0.002	S5_74	0.005
S5_28	0.002	S5_75	0.007
S5_31	0.005	S5_78	-0.008
S5_32	0.005	S5_82	0.005
S5_35	0.005	S5_83	0.006
S5_37	0.002	S5_85	0.003

Table 9

A table containing the classifications of importance for each variable as determined by the Boruta() function.

Final Classification of Importance	Variables
Confirmed Important	MQ12, MQ13, MQ16, S5_3, S5_34, S5_35, S5_39, S5_40, S5_53, S5_56, S5_57, S5_59, S5_60, S5_67, S5_75
Rejected	MQ1, MQ2, MQ3, MQ4, MQ5, MQ6, MQ7, MQ8, MQ10, MQ11, MQ14, MQ15, MQ18, MQ20, S5_1, S5_4, S5_6, S5_7, S5_8, S5_9, S5_10, S5_11, S5_12, S5_13, S5_14, S5_15, S5_16, S5_17, S5_19, S5_20, S5_23, S5_24, S5_26, S5_28, S5_29, S5_30, S5_32, S5_33, S5_36, S5_37, S5_41, S5_42, S5_43, S5_44, S5_45, S5_46, S5_47, S5_48, S5_49, S5_50, S5_51, S5_52, S5_54, S5_55, S5_58, S5_64, S5_65, S5_66, S5_68, S5_70, S5_71, S5_72, S5_73, S5_74, S5_76, S5_77, S5_78, S5_79, S5_80, S5_82, S5_83, S5_84, S5_86, S5_87
Tentative	MQ17, MQ19, S5_2, S5_5, S5_18, S5_21, S5_22, S5_25, S5_27, S5_31, S5_38, S5_61, S5_62, S5_63, S5_69, S5_81, S5_85

Table 10

A table containing the predictive accuracy values obtained by all models built in this tutorial paper. Methods are listed such that the accuracy values are ordered from least accurate to most accurate. Significance is determined relative to the previous model (i.e., Elastic SCAD SVM was determined to have a statistically significant better accuracy than Elastic Net non-biased) according to a McNemar's Chi-squared test with continuity correction.

Method	Cross-validated Accuracy
Elastic Net non-biased	0.797
Elastic SCAD SVM	0.828**
LASSO	0.859**
Elastic Net	0.875
Boruta	0.875
LASSO non-biased	0.891
GA	1***

* $p < .05$, ** $p < .01$, *** $p < .0001$

Figure 1

The leftmost graph illustrates an overfit model on a small amount of data. We then see the influence of overfitting with the introduction of new data in the center graph. Lastly, in the rightmost graph, a new model was fit using both the old and new data.

Figure 2

Cross-validated estimate of the mean squared prediction error for LASSO as a function of the log λ. The upper axis indicates the number of non-zero coefficients in the regression model at the given log λ. The dashed vertical line illustrates the location of the CV minimum and the one standard error rule locations for λ.

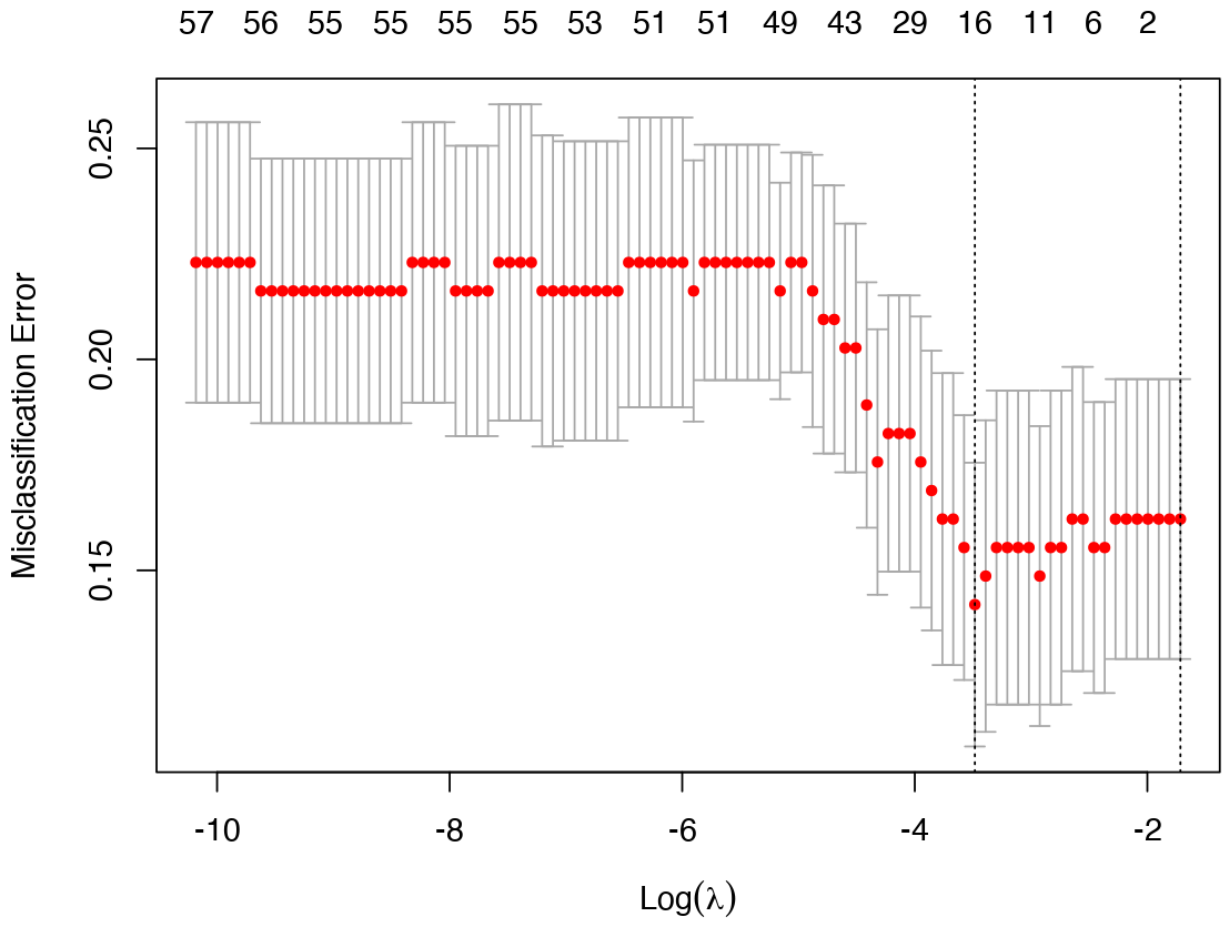

Figure 3

The basic algorithmic steps of the Genetic Algorithm.

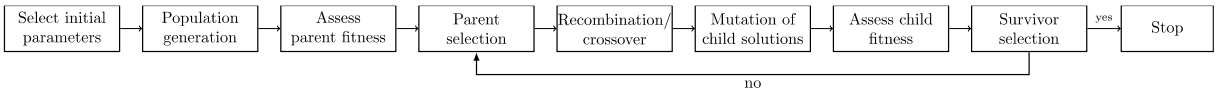

Figure 4

An example of a decision tree built on the misophonia data using the ctree() function.

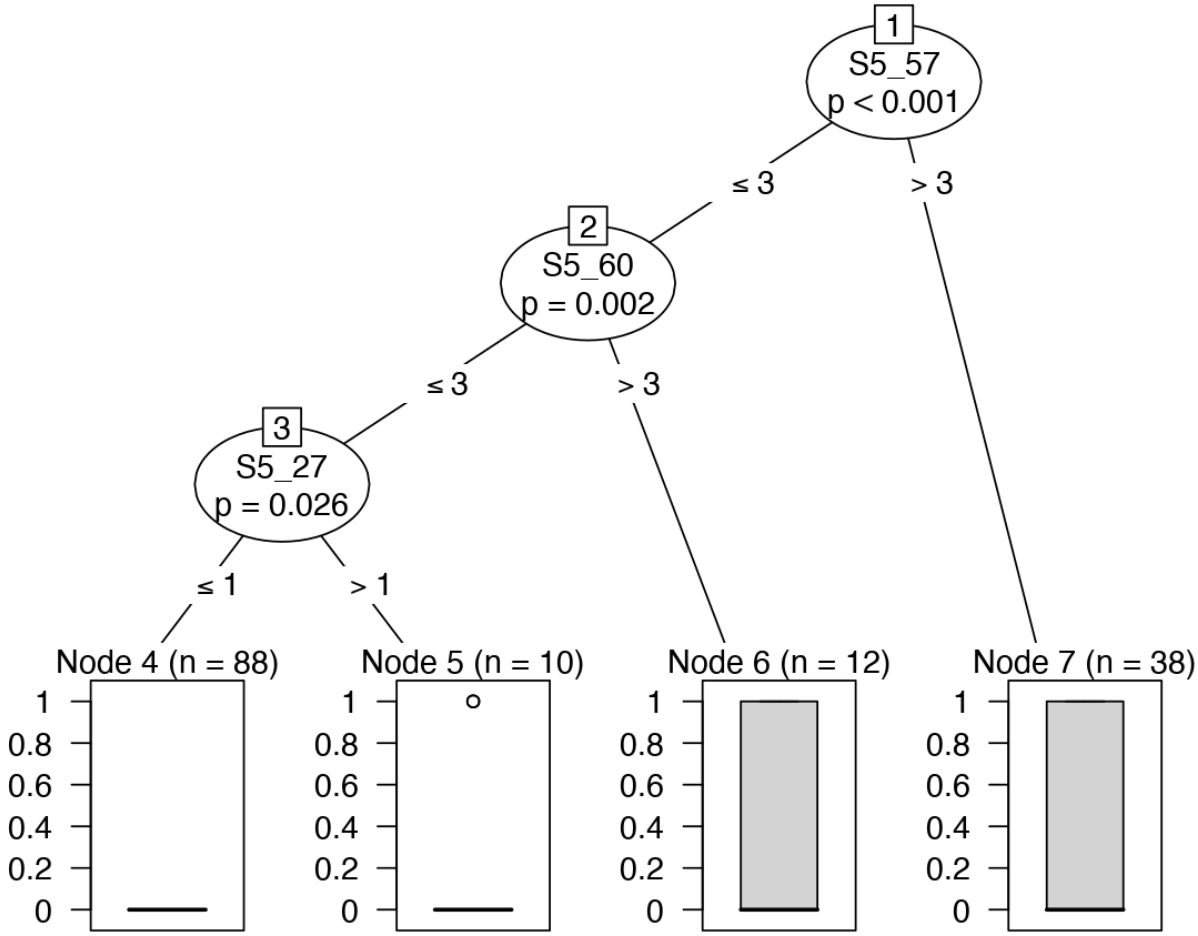

Figure 5

A plot containing the Z-score transformed estimates of variable importance scores for each variable in the Boruta() model. Blue boxplots correspond to minimal, average, and maximum Z-scores of a shadow attribute. Red and green boxplots represent Z-scores of rejected and confirmed attributes respectively.