13

SMALL SAMPLE META-ANALYSES

Exploring heterogeneity using MetaForest

Caspar J. van Lissa

DEPARTMENT OF METHODOLOGY AND STATISTICS, UTRECHT UNIVERSITY, UTRECHT, THE NETHERLANDS

Introduction

Meta-analysis is the act of statistically summarizing the findings of several studies on a single topic (Borenstein, Hedges, Higgins, & Rothstein, 2009). Some consider meta-analyses to be the golden standard of scientific evidence (Crocetti, 2016). This reputation is not entirely deserved, however, as meta-analysis comes with its own pitfalls. One of these is that meta-analyses often present a small sample problem. The fact that each of the studies included in the meta-analysis is based on a larger sample of participants does not mean that the problem of small sample sizes is any less relevant than in primary research. Particularly in the social sciences, the number of studies on any topic is typically low, because conducting research is cost- and time-intensive. In an investigation of 705 psychological meta-analyses, the median number of studies was 12 (Van Erp, Verhagen, Grasman, & Wagenmakers, 2017), and in 14 meta-analyses from education science, the median number of studies was 44 (De Jonge & Jak, 2018). Small sample sizes thus appear to be the rule, rather than the exception.

The issue of small sample sizes is compounded by a related challenge: Between-studies heterogeneity (Higgins & Thompson, 2002). Differences between the studies can introduce heterogeneity in the effect sizes found. These two problems are related, because small samples have limited statistical power to adequately account for sources of between-studies heterogeneity. In this chapter, I discuss how the related problems of small sample sizes and between-studies heterogeneity can be overcome using MetaForest: A machine-learning-based approach to identify relevant moderators in meta-analysis (Van Lissa, 2017). After presenting the general principles underlying this technique, I provide a tutorial example for conducting a small sample meta-analysis, using the R package `metaforest` (Van Lissa, 2018).

Models for meta-analysis

Classic meta-analysis can be conceptualized as a weighted average of study effect sizes, where studies with a larger sample accrue greater weight. The simplest statistical model used to assign these weights is the so-called fixed-effect model (Hedges & Vevea, 1998). The fixed-effect model does not account for between-studies heterogeneity. This model assumes that all studies tap into one true effect size (T), and that any differences between observed effect sizes are due to sampling error. This assumption is probably valid when the included studies are very close replications.

In most cases, however, this assumption is too restrictive, and we assume that some between-studies heterogeneity exists. If we can assume that the heterogeneity is "random", or normally distributed, we can use a random-effects model, which assumes that each study taps into an underlying (normal) distribution of true effect sizes (Hedges & Vevea, 1998). The random-effects model estimates the mean and standard deviation of this distribution. This model is the appropriate choice if the studies are similar (e.g., replications from different labs), but some small unknown random differences might have crept in. In the random-effects model, the weight accorded to each effect size is no longer purely based on its sample size – it is also based on the estimated between-studies heterogeneity. In the hypothetical case that the between-studies heterogeneity is estimated to be zero, the weights are the same as in the fixed-effect model. When heterogeneity is larger, however, the study weights are adjusted to be more equal, because each study now conveys some information about a different area of the underlying distribution of effect sizes. If the between-studies heterogeneity would be huge, all studies would be weighted equally.

Between-studies heterogeneity

A common application of meta-analysis in the social sciences is to summarize a diverse body of literature on a specific topic. The literature typically covers similar research questions, investigated in different laboratories, using different methods, instruments, and samples (Maxwell, Lau, & Howard, 2015). The assumption of the random-effects model, that there is one underlying normal distribution of true effect sizes, likely breaks down in such cases (Hedges & Vevea, 1998), because these between-studies differences might introduce heterogeneity in the effect sizes.

Researchers can account for between-studies differences by coding them as *moderator variables*, and controlling for their influence using meta-regression (Higgins & Thompson, 2004). Similar to classic regression, meta-regression posits that the outcome – in this case, the effect size of a study – is a function of the value of the moderators for that study. Both the fixed-effects and random-effects model can be extended to meta-regression. The advantage of coding between-studies differences as moderators, rather than using them as exclusion criteria, is that all studies can be included, as long as any differences are controlled for using meta-regression.

Too many moderators

Like any regression-based technique, meta-regression requires relatively many cases per parameter (Guolo & Varin, 2017). But in heterogeneous fields of research, there are often many potential moderators. We typically do not know beforehand which moderators will affect the effect size found. If we just include all moderators in a meta-regression, we risk overfitting the data (Higgins & Thompson, 2004). *Overfitting* means that the model fits the observed data very well, but does not generalize to new data, or the population (Hastie, Tibshirani, & Friedman, 2009). This is because it captures noise in the data, not just genuine effects. The more moderators are included, the more prone a model becomes to overfitting.

The problem of small samples is compounded by the existence of between-studies differences that could potentially influence the effect size of a study. The more potential moderators, the larger the sample that would be required to adequately account for their influence. Moreover, these two problems tend to go hand in hand. When there is a small body of literature on a given topic, it tends to be comprised of idiosyncratic studies.

How to deal with moderators?

Based on my experience as a statistical consultant, I have found that the question of how to deal with moderators is one of the most common challenges researchers face when conducting a meta-analysis. One common approach appears to be to diligently code moderators, but then omit them from the analysis. Most data sets I have requested from authors of published meta-analyses contained more moderators than discussed in the paper. In one extreme case, a meta-analysis of 180 studies reported a single moderator, whereas the raw data set contained over 190 moderators – more variables than studies. Of course, the problem of having many potentially relevant moderators is not resolved by failing to report them. They will introduce between-studies heterogeneity regardless. It is unlikely that this selective reporting is ill-intentioned, as I have found most authors of meta-analyses to be very willing to share their data. A more likely explanation is that authors lack concrete guidelines on how to whittle down the list of potential moderators to a manageable number.

A second common practice appears to be to preselect moderators using univariate meta-regressions, and to retain those whose p-value falls below a certain threshold. This is problematic, as (1) the p-value is not a measure of variable importance, (2) repeated tests inflate the risk of false positive results, and (3) coefficients in the model are interdependent, and omitting one moderator can influence the effect of others. Another approach is to run a model including all moderators, and then eliminate non-significant ones. This is problematic for all but the second aforementioned reasons. Additionally, when the number of

moderators is relatively large compared to the number of studies included, the risk of overfitting increases.

A method for exploratory moderator selection

What is needed is a technique that can explore between-studies heterogeneity and perform variable selection, identifying relevant moderators from a larger set of candidates, without succumbing to overfitting. The recently developed MetaForest algorithm meets these requirements (Van Lissa, 2017). MetaForest is an adaptation of the random forest algorithm (Breiman, 2001; Strobl, Malley, & Tutz, 2009) for meta-analysis. Random forests are a powerful machine learning algorithm for regression problems, with several advantages over linear regression. First, random forests are robust to overfitting. Second, they are non-parametric, and can inherently capture non-linear relationships between the moderator and effect size, or even complex, higher-order interactions between moderators. Third, they perform variable selection, identifying which moderators contribute most strongly to the effect size found.

Understanding random forests

The *random forest* algorithm combines many *tree models* (Hastie et al., 2009). A tree model can be conceptualized as a decision tree, or a flowchart: Starting with the full data set, the model splits the data into two groups. The splitting decision is based on the moderator variables; the model finds a moderator variable, and the value on that variable, along which to split the data set. It chooses the moderator and value that result in the most homogenous post-split groups possible. This process is repeated for each post-split group; over and over again, until a stopping criterion is reached. Usually, the algorithm is stopped when the post-split groups contain a minimum number of cases.

One advantage of regression trees is that it does not matter if the number of moderators is large relative to the sample size, or even exceeds it. Second, trees are non-parametric; they do not assume normally distributed residuals or linearity, and intrinsically capture non-linear effects and interactions. These are substantial advantages when performing meta-analysis on a heterogeneous body of literature. Single regression trees also have a limitation, however, which is that they are extremely prone to overfitting. They will simply capture all patterns in the data, both genuine effects and random noise (Hastie et al., 2009).

Random forests overcome this limitation of single regression trees. First, many different bootstrap samples are drawn (e.g., 1,000). Then, a single tree is grown on each bootstrap sample. To ensure that each tree learns something unique from the data, only a small random selection of moderators is made available to choose from at each splitting point. Finally, the predictions of all tree models are averaged. This renders random forests robust to overfitting:

Because each tree captures some of the true patterns in the data, and overfits some random noise that is only present in its bootstrap sample, overfitting cancels out on aggregate. Random forests also make better predictions: Where single trees predict a fixed value for each "group" they identify in the data, random forests average the predictions of many trees, which leads to smoother prediction curves.

An earlier chapter pointed out that bootstrapped confidence intervals for hypothesis testing are not valid as a small sample technique (see also Chapter 18 by Hox). As samples get smaller, their representativeness of the population decreases. Consequently, bootstrap resampling will be less likely to yield an accurate approximation of the sampling distribution. The purpose of bootstrapping in random forests is different from hypothesis testing, however: It aims to ensure that every tree model explores some unique aspects of the *data at hand*. Thus, concerns regarding bootstrapped hypothesis tests are not directly relevant here.

Meta-analytic random forests

To render random forests suitable for meta-analysis, a weighting scheme is applied to the bootstrap sampling, which means that more precise studies exert greater influence in the model building stage (Van Lissa, 2017). These weights can be uniform (each study has equal probability of being selected into the bootstrap sample), fixed-effects-based (studies with smaller sampling variance have a larger probability of being selected), or random-effects-based (studies with smaller sampling variance have a larger probability of being selected, but this advantage is diminished as the amount of between-studies heterogeneity increases). Internally, `MetaForest` relies on the ranger R package; a fast implementation of the random forests in `C++` (Wright & Ziegler, 2015).

Tuning parameters

Like many machine learning algorithms, random forests have several "tuning parameters": Settings that might influence the results of the analysis, and whose optimal values must be determined empirically. The first is the number of candidate variables considered at each split of each tree. The second is the minimum number of cases that must remain in a post-split group within each tree. The third is unique to MetaForest; namely, the type of weights (uniform, fixed-, or random-effects). The optimal values for these tuning parameters are commonly determined using cross-validation (Hastie et al., 2009). Cross-validation means splitting the data set many times; for example, into 10 equal parts. Then, predictions are made for each of the parts of the data, using a model estimated on all of the other parts. This process is conducted for all possible combinations of tuning parameters. The values of tuning parameters that result in the lowest

cross-validated prediction error are used for the final model. For cross-validation, `MetaForest` relies on the well-known machine learning R package caret (Kuhn, 2008).

Understanding the output

The output of a MetaForest analysis is somewhat different from what researchers schooled in the general linear model might be familiar with. Three parts of the output, in particular, warrant further clarification.

Predictive performance

Just like regression, random forests offer a measure of explained variance similar to R^2. Whereas R^2 refers to the variance explained in the data *used to estimate the model*, random forests provide an estimate of how much variance the model would explain in a *new data set* (Hastie et al., 2009). This distinction between "retrodictive" and "predictive" performance is important: The retrodictive R^2 increases with every moderator added to the model, even when the model is overfit. However, such an overfit model would make terrible predictions for new data.

Random forests provides an estimate of predictive performance, called R^2_{oob} (Breiman, 2001). The subscript *oob* stands for "out-of-bag" and refers to the way this estimate is obtained: By predicting each case in the data set from those trees that were trained on bootstrap samples *not* containing that case. A second estimate of predictive R^2, R^2_{cv}, is obtained during cross-validation (*cv*), by predicting cases not used to estimate the model. Predictive R^2 becomes negative when a model is overfit, because the model makes worse predictions than the mean for new data. A negative R^2_{oob} or R^2_{cv} can thus be interpreted as a sign of overfitting. Positive values estimate how well the model will predict the effect sizes of new studies.

Variable importance

The second relevant type of output are variable importance metrics, which quantify the relative importance of each moderator in predicting the effect size. These metrics are analogous in function to the (absolute) standardized regression coefficients (β^z) in regression: They reflect the strength of each moderator's relationship with the outcome on a common metric. However, whereas betas reflect linear, univariate, partial relationships, MetaForest's variable importance metrics reflect each moderator's contribution to the predictive power of the final model across all linear-, non-linear-, and interaction effects. Variable importance is estimated by randomly permuting, or shuffling, the values of a moderator, thereby annulling any relationship that moderator had with the outcome, and then observing how much the predictive

performance of the final model drops. If performance drops a lot, the moderator must have been important. Variable importance can be negative when a moderator is weakly associated with the outcome, and random shuffling coincidentally strengthens the relationship. Such moderators can be dropped from the model. In the R package `metaforest`, variable importance can be plotted using the `VarImpPlot()` function.

Effects of moderators

Random forests are not a black box: Partial dependence plots can be used to visualize the shape of the marginal relationship of each moderator to the effect size, averaging over all values of the other moderators. Researchers commonly inspect only univariate marginal dependence plots. Exploring all possible higher-order interactions swiftly becomes unmanageable; with just 10 moderators, the number of bivariate interactions is 45, and the number of trivariate interactions is 120. In order to plot bivariate interactions with a specific moderator of theoretical relevance, you can use the `PartialDependence()` function in conjunction with the moderator argument.

Accounting for dependent data

Studies often report multiple effect sizes; for example, because several relevant outcomes have been measured. In traditional meta-analysis, one might account for this dependency in the data by using a multilevel analysis (Van Den Noortgate, López-López, Marín-Martínez, & Sánchez-Meca, 2015). With random forests, dependent data leads to an under-estimation of the aforementioned out-of-bag error, which is used to calculate R^2_{oob} and variable importance (Janitza, Celik, & Boulesteix, 2016). If the model has been estimated based on some effect sizes from one study, it will likely have an advantage at predicting other effect sizes from the same study. Thus, the out-of-bag error will be misleadingly small, and hence, the R^2_{oob} will be positively biased. In MetaForest, this problem is overcome by using clustered bootstrap sampling, as proposed by Janitza et al. (2016).

Suitability for small samples

MetaForest has been evaluated in simulation studies, in terms of its predictive performance, power, and ability to identify relevant versus irrelevant moderators (Van Lissa, 2017). The full syntax of these simulations is available at osf.io /khjgb/. To determine practical guidelines for the usage of MetaForest with small samples, it is instructive to examine under what conditions a model estimated using MetaForest predicts new data with greater accuracy than the mean at least 80% of the time. The simulation studies indicated that MetaForest met this criterion in most cases with as few as 20 included studies, except when the effect size of moderators was small (data were simulated based on a linear model, with an effect size of .2), and residual heterogeneity was very large (as compared

to values commonly reported in psychological meta-analyses; Van Erp et al. (2017). This suggests that MetaForest is suitable as a small sample solution.

In applied research, the true effect size and residual heterogeneity are unknown. So how do you determine whether MetaForest has detected any reliable effects of moderators? One possibility is to adapt the published syntax of these simulation studies to conduct a custom-made power analysis. Second, with a larger data set, one could set aside part of the data, a "test set". One could then estimate the model on the remaining part of the data, the "training set", and compute a predictive R^2 on the test set; R^2_{test}. With small samples, however, this approach is problematic, because what little data there is should go into the main analysis. Consequently, the most feasible small sample solution might be to examine the R^2_{oob} or R^2_{cv}, as alternatives to the R^2_{test}.

Feature pre-selection

One pitfall with random forests is that they can overfit if a data set contains many *irrelevant* predictors; moderators unrelated to the outcome. Recall that at every split of each tree, a random subset of moderators is made available to choose from. If there are many "noise" predictors, the model will occasionally be forced to select among only irrelevant predictors. This risk is increased when the sample is small, and there are relatively many predictors relative to cases. Thus, it might be desirable to eliminate some noise variables. As mentioned before, noise variables can be identified by their negative variable importance. However, in a small model with many noise variables, these variable importance metrics can vary substantially when re-running the analysis, due to Monte Carlo error introduced by the random aspects of the analysis – Bootstrap sampling, and the random subset of variables considered at each split. Consequently, it can be useful to replicate the analysis, visualize the distribution of variable importance metrics, and filter out variables that have a (mostly) negative variable importance across replications. This is accomplished by using the `preselect()` function, which can implement a simple replication of the analysis, or a bootstrapped replication, or a recursive selection algorithm.

Using MetaForest for small samples

To illustrate how to use MetaForest to identify relevant moderators in a small sample meta-analysis, I will re-analyze the published work of Fukkink and Lont (2007), who have graciously shared their data. The authors examined the effectiveness of training on the competency of childcare providers. The sample is small, consisting of 78 effect sizes derived from 17 unique samples. Exploratory moderator analysis was an explicit goal of the original work: "The first explorative question concerns the study characteristics that are associated with experimental results." Data for this tutorial are included in the `metaforest` package.

```
# Install metaforest. This needs to be done only once.
install.packages("metaforest")
# Load the metaforest package
library(metaforest)
# Assign the fukkink_lont data to an object called "data"
data <- fukkink_lont
# Set a seed for the random number generator,
# so analyses can be replicated exactly.
set.seed(62)
```

For any random forest model, it is important to check whether the model converges. Convergence is indicated by stabilization of the cumulative mean squared out-of-bag prediction error (MSE_{oob}), as a function of the number of trees in the model. We run the analysis once with a very high number of trees, and pick a smaller number of trees, at which the model is also seen to have converged, to speed up computationally heavy steps, such as replication and model tuning. We re-examine convergence for the final model.

```
# Run model with many trees to check convergence
check_conv <- MetaForest(yi~.,
                         data = data,
                         study = "id_exp",
                         whichweights = "random",
                         num.trees = 20000)
# Plot convergence trajectory
plot(check_conv)
```

This model has converged with approximately 10,000 trees (Figure 13.1). We now apply moderator pre-selection with this number of trees, using the `preselect()` function. The `"recursive"` pre-selection algorithm conducts one MetaForest analysis, drops the moderator with the most negative variable importance, and then reruns the analysis, until all remaining variables have positive importance. This recursive algorithm is replicated 100-fold. Using `preselect_vars()`, we retain only those moderators for which a 50% percentile interval of the variable importance metrics does not include zero (variable importance is counted as zero when a moderator is not included in the final step of the recursive algorithm). The results of this preselection can be plotted using plot () (see Figure 13.2).

```
# Model with 10000 trees for replication
mf_rep <- MetaForest(yi~.,
                         data = data,
                         study = "id_exp",
                         whichweights = "random",
                         num.trees = 10000)
# Recursive preselection
preselected <- preselect(mf_rep,
                         replications = 100,
                         algorithm = "recursive")
# Plot results
plot(preselected)
```

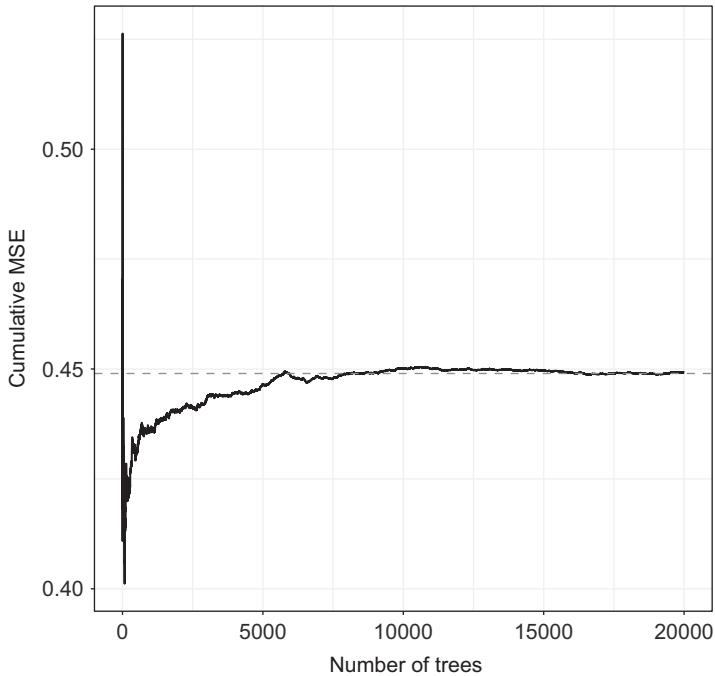

FIGURE 13.1 Convergence plot

```
# Retain moderators with positive variable importance in more than
# 50% of replications
retain_mods <- preselect_vars(preselected, cutoff = .5)
```

Next, we tune the model using the R package `caret`, which offers a uniform workflow for any machine learning task. The function `ModelInfo_mf()` tells `caret` how to tune a MetaForest analysis. As tuning parameters, we consider all three types of weights (uniform, fixed-, and random-effects), the number of candidate variables at each split from 2–6, and a minimum node size from 2–6. We select the model with smallest root mean squared prediction error (RMSE) as the final model, based on 10-fold clustered cross-validation. Clustered cross-validation means that effect sizes from the same study are always included in the same fold, to account for the dependency in the data. Note that the number of folds cannot exceed the number of clusters in the data. Moreover, if the number of clusters is very small, one might have to resort to specifying the same number of folds as clusters. Model tuning is computationally intensive and might take a long time.

```
# Load caret
library(caret)
# Set up 10-fold clustered CV
grouped_cv <- trainControl(method = "cv",
                           index = groupKFold(data$id_exp, k = 10))
```

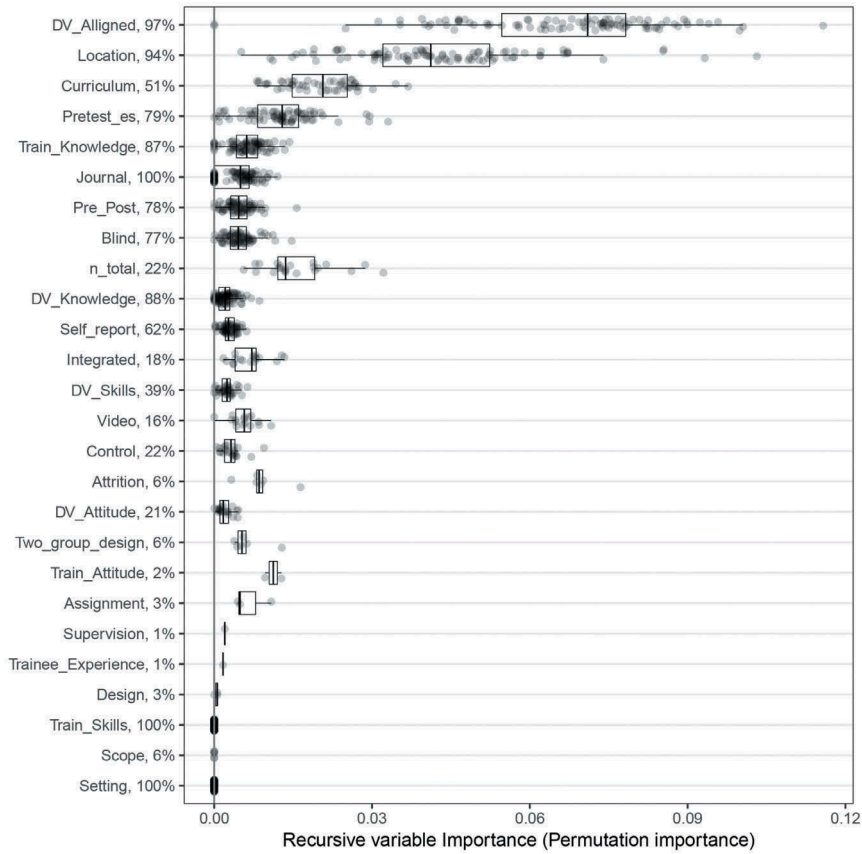

FIGURE 13.2 Replicated variable importance for moderator pre-selection

```
# Set up a tuning grid
tuning_grid <- expand.grid(whichweights = c("random", "fixed", "unif"),
                           mtry = 2:6,
                           min.node.size = 2:6)

# X should contain only retained moderators, clustering variable, and vi
X <- data[, c("id_exp", "vi", retain_mods)]

# Train the model
mf_cv <- train(y = data$yi,
               x = X,
               study = "id_exp", # Name of the clustering variable
               method = ModelInfo_mf(),
               trControl = grouped_cv,
               tuneGrid = tuning_grid,
               num.trees = 10000)
# Extract R^2_cvVan Lissa,
r2_cv <- mf_cv$results$Rsquared[which.min(mf_cv$results$RMSE)]
```

Based on the root mean squared error, the best combination of tuning parameters were uniform weights, with four candidate variables per split, and a minimum of two cases per terminal node. The object returned by train already contains the final model, estimated with the best combination of tuning parameters.

```
# Extract final model
final <- mf_cv$finalModel
# Extract R^2_oob from the final model
r2_oob <- final$forest$r.squared
# Plot convergence
plot(final)
```

We can conclude that the model has converged (Figure 13.3), and has a positive estimate of explained variance in new data, $R^2_{oob} = 0.13$, $R^2_{cv} = 0.48$. Now, we proceed to interpreting the moderator effects, by examining variable importance (Figure 13.4), and partial dependence plots (Figure 13.5).

```
# Plot variable importance
VarImpPlot(final)
# Sort the variable names by importance
ordered_vars <- names(final$forest$variable.importance)[
  order(final$forest$variable.importance, decreasing = TRUE)]
```

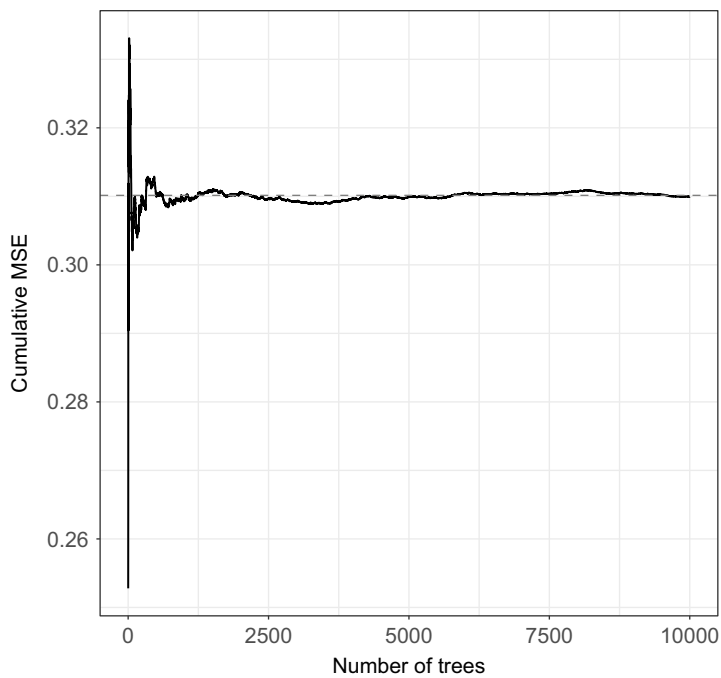

FIGURE 13.3 Convergence plot for final model

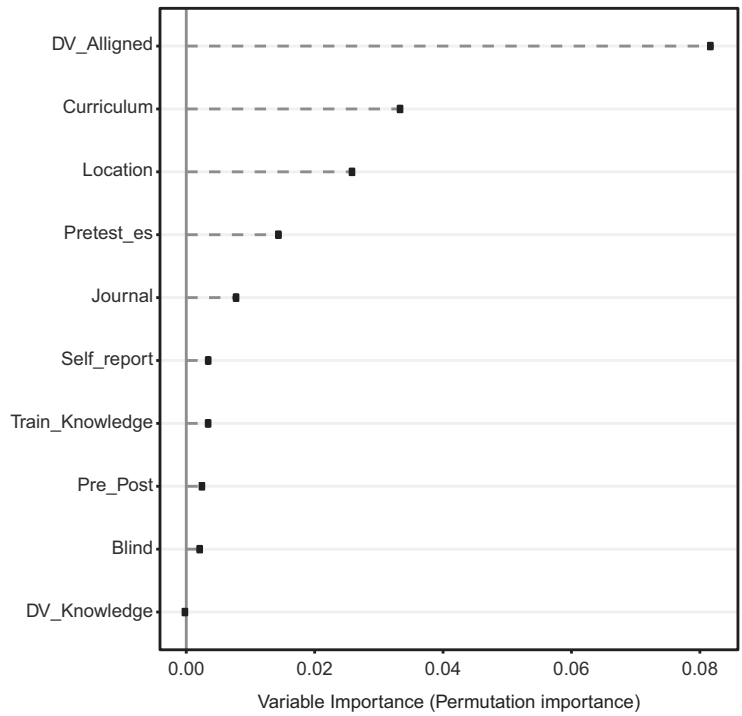

FIGURE 13.4 Variable importance plot

```
# Plot partial dependence
PartialDependence(final, vars = ordered_vars,
                  rawdata = TRUE, pi = .95)
```

We cannot conclude whether any of these findings are "significant" (except perhaps by bootstrapping the entire analysis). However, the `PartialDependence()` function has two settings that help visualize the "importance" of a finding: `rawdata`, which plots the weighted raw data (studies with larger weights are plotted with a larger point size), thereby visualizing the variance around the mean prediction, and `pi`, which plots a (e.g., 95%) percentile interval of the predictions of individual trees in the model. This is not the same as a confidence interval, but it does show how variable or stable the model predictions are.

The analysis has revealed, for example, that effect sizes tend to be stronger when the dependent variable is in line with the content of the intervention, and that single-site training interventions tend to have bigger effect sizes (Figure 13.4). Because these variables are binary, their effects could also be parsimoniously modeled by a linear regression analysis. Indeed, the original paper reported significant effects for these variables. Non-linear effects, on the other hand, are more easily overlooked in a linear meta-regression.

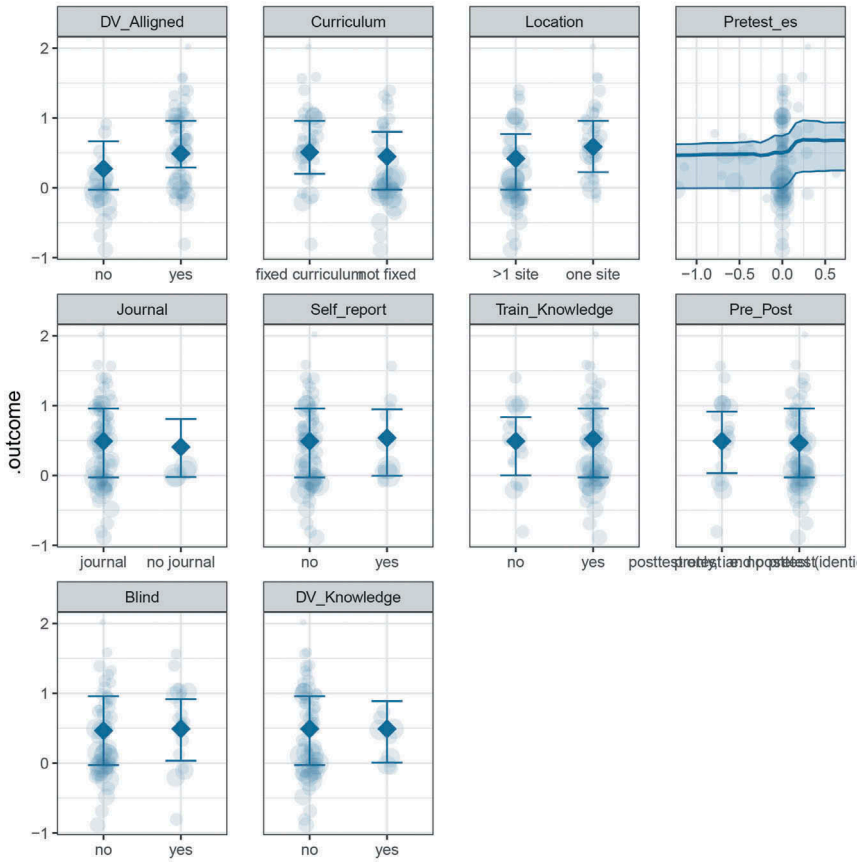

FIGURE 13.5 Marginal relationship of moderators with effect size

This exploratory moderator analysis could be followed with meta-regression, focusing only on the relevant moderators. The binary predictors could be straightforwardly included. For the continuous variables, one might consider a piecewise linear approach: Creating dummy variables at the inflection points identified from the partial dependence plots, and then interacting these dummy variables with the continuous variable itself. However, the exploratory nature of this follow-up analysis should always be emphasized; it is merely a way to look at the same results from the familiar linear regression framework.

What to report

The preceding paragraphs offer a step-by-step instruction on how one might go about conducting a MetaForest analysis on a small sample meta-analytic data set. One could simply apply these steps to a different data set. If readers are

concerned with the amount of space required to report and explain this type of analysis in a journal whose readership might be relatively unfamiliar with the machine learning approach, then one might simply report the analysis summary, and cite appropriate publications for MetaForest (Van Lissa, 2017), and random forests in general (e.g., Strobl et al., 2009). Because it is essential that the analysis process is reproducible and transparent, the annotated syntax – and, preferably, the data – can be published as supplementary material on the Open Science Framework (www.osf.io), and referred to in the paper. For example:

> We conducted an exploratory search for relevant moderators using MetaForest: a machine-learning-based approach to meta-analysis, using the random forests algorithm (Van Lissa, 2017). Full syntax of this analysis is available on the Open Science Framework, DOI:10.17605/OSF.IO/XXXXX. To weed out irrelevant moderators, we used 100-fold replicated feature selection, and retained only moderators with positive variable importance in > 10% of replications. The main analysis consisted of 10.000 regression trees with fixed-effect weights, four candidate variables per split, and a minimum of three cases per terminal node. The final model had positive estimates of explained variance in new data, $R^2_{oob} = 0.13$, $R^2_{cv} = 0.48$. The relative importance of included moderators is displayed in Figure X. The shape of each moderator's marginal relationship to the effect size, averaging over all values of all other moderators, is illustrated in Figure XX.

Several published studies illustrate ways to apply and report MetaForest analyses. For example, Curry et al. (2018) used MetaForest to examine moderators of the effect of acts of kindness on well-being (full syntax and data available at github.com/cjvanlissa/kindness_meta-analysis). Second, Bonapersona et al. (in press) used MetaForest to identify moderators of the effect of early life adversity on the behavioral phenotype of animal models, with full syntax and data available at osf.io/ra947/. Third, Gao, Yao, and Feldman (2018) used MetaForest to examine moderators of the "mere ownership" effect.

Final thoughts

MetaForest is a helpful solution to detect relevant moderators in meta-analysis, even for small samples. Its main advantages over classic meta-regression are that it is robust to overfitting, captures non-linear effects and interactions, and is robust even when there are many moderators relative to cases. One remaining concern, which cannot be addressed by any statistical solution, is the generalizability of these findings to genuinely new data. When the sample of studies is small, it is unlikely to be representative of the entire "population" of potential studies that could have been conducted. Machine learning techniques, such as MetaForest, aim to optimize a model's performance in "new data" – but the

estimates of performance in "new data", based on bootstrap aggregation and cross-validation, are still conditional on the present sample.

What implications might this have? To understand the problem, we might imagine conducting a primary study on the link between father involvement and child well-being, and drawing a sample by selecting one citizen of every country in the European Union. Whether this study will generate any reliable insights that generalize beyond this selective sample depends, in part, on the strength of the effect, and the heterogeneity between our different Europeans. But it also depends on the universality of the phenomenon under study. If father involvement benefits children all around the world, we will be more likely to detect an effect, even in such a heterogeneous sample. If the association is not universal, it might be moderated, and we can measure these moderators and use an inductive approach like MetaForest to identify which ones make a difference.

Another remaining concern is that the cumulative nature of science means that researchers are typically building upon the work of their predecessors. Consequently, we might ask whether it is ever possible for a body of literature to be considered a random sample of the population of all possible studies that "could have been". If the answer is no, then it would be prudent to consider every meta-analysis to be, to some extent, merely a descriptive instrument; a quantitative summary of the published literature.

References

Bonapersona, V., Kentrop, J., Van Lissa, C. J., Van der Veen, R., Joels, M., & Sarabdjitsingh, R. A. (in press). *The behavioral phenotype of early life adversity: A 3-level meta-analysis of rodent studies: Supplemental material*. bioRxiv.

Borenstein, M., Hedges, L. V., Higgins, J. P. T., & Rothstein, H. R. (2009). *Introduction to meta-analysis*. Chichester: John Wiley & Sons.

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

Crocetti, E. (2016). Systematic reviews with meta-analysis: Why, when, and how? *Emerging Adulthood*, *4*(1), 3–18.

Curry, O. S., Rowland, L. A., Van Lissa, C. J., Zlotowitz, S., McAlaney, J., & Whitehouse, H. (2018). Happy to help? A systematic review and meta-analysis of the effects of performing acts of kindness on the well-being of the actor. *Journal of Experimental Social Psychology*, *76*, 320–329.

De Jonge, H., & Jak, S. (2018). *A meta-meta-analysis: identifying typical conditions of meta-analyses in educational research*. Retrieved from https://osf.io/zau68/.

Fukkink, R. G., & Lont, A. (2007). Does training matter? A meta-analysis and review of caregiver training studies. *Early Childhood Research Quarterly*, *22*(3), 294–311.

Gao, Y., Yao, D., & Feldman, G. (2018). *Owning leads to valuing: meta-analysis of the mere ownership effect*. Unpublished. doi: 10.13140/RG.2.2.13568.33287/1.

Guolo, A., & Varin, C. (2017). Random-effects meta-analysis: The number of studies matters. *Statistical Methods in Medical Research*, *26*(3), 1500–1518.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Berlin: Springer.

Hedges, L. V., & Vevea, J. L. (1998). Fixed- and random-effects models in meta-analysis. *Psychological Methods*, *3*(4), 486–504.

Higgins, J. P. T., & Thompson, S. G. (2002). Quantifying heterogeneity in a meta-analysis. *Statistics in Medicine*, *21*(11), 1539–1558.

Higgins, J. P. T., & Thompson, S. G. (2004). Controlling the risk of spurious findings from meta-regression. *Statistics in Medicine*, *23*(11), 1663–1682.

Janitza, S., Celik, E., & Boulesteix, A.-L. (2016). A computationally fast variable importance test for random forests for high-dimensional data. *Advances in Data Analysis and Classification*, *12*(4), 1–31.

Kuhn, M. (2008). Building predictive models in R using the caret package. *Journal of Statistical Software, Articles*, *28*(5), 1–26.

Maxwell, S. E., Lau, M. Y., & Howard, G. S. (2015). Is psychology suffering from a replication crisis? What does "failure to replicate" really mean? *American Psychologist*, *70*(6), 487–498.

Strobl, C., Malley, J., & Tutz, G. (2009). An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological Methods*, *14*(4), 323–348.

Van Den Noortgate, W., López-López, J. A., Marín-Martínez, F., & Sánchez-Meca, J. (2015). Meta-analysis of multiple outcomes: A multilevel approach. *Behavior Research Methods*, *47*(4), 1274–1294.

Van Erp, S., Verhagen, J., Grasman, R. P. P. P., & Wagenmakers, E.-J. (2017). Estimates of between-study heterogeneity for 705 meta-analyses reported in psychological bulletin from 1990–2013. *Journal of Open Psychology Data*, *5*, 1.

Van Lissa, C. J. (2017). MetaForest: Exploring heterogeneity in meta-analysis using random forests. Open Science Framework. doi:10.17605/OSF.IO/KHJGB.

Van Lissa, C. J. (2018). *Metaforest: exploring heterogeneity in meta-analysis using random forests (version 0.1.2) [R-package]*. Retrieved from https://CRAN.R-project.org/package=metaforest.

Wright, M. N., & Ziegler, A. (2015). Ranger: A fast implementation of random forests for high-dimensional data in C++ and R. *arXiv:1508.04409 [stat]*.