Stefan Th. Gries*

On classification trees and random forests in corpus linguistics: Some words of caution and suggestions for improvement

https://doi.org/10.1515/cllt-2018-0078

Abstract: This paper is a discussion of methodological problems that (can) arise in the analysis of multifactorial data analyzed with tree-based or forest-based classifiers in (corpus) linguistics. I showcase a data set that highlights where such methods can fail at providing optimal results and then discuss solutions to this problem as well as the interpretation of random forests more generally.

Keywords: classification trees, random forests, regression modeling

1 Introduction

1.1 The use of tree-based methods in current (corpus) linguistics

Over the last 20 or so years, multifactorial modeling has taken much of corpus linguistics by storm. Compared to, say, the 1980s or even the 1990s, there is now a huge and constantly growing number of studies in corpus linguistics that tackle phenomena of interest with methods that allow researchers to study the effect of multiple independent variables, or predictors, on the dependent variable, or response, of interest. The by far most widely used statistical tool in this respect is probably that of generalized linear (regression) modeling, either directly or, as in the case of sociolinguistic Varbrul applications, indirectly.

However, over the last few years, many researchers have also realized that generalized linear models or their extension to generalized linear mixed-effects models can run into problems especially when applied to observational data such as corpus data. This is because, unlike nicely balanced experimental data, observational corpus data exhibit a variety of characteristics which make regression modeling hard:

***Corresponding author: Stefan Th. Gries,** University of California, Santa Barbara, CA, USA; Justus Liebig University Giessen, Giessen, Germany, E-mail: stgries@gmail.com

- The data are often (extremely) Zipfian distributed (Ellis et al. 2016), which means that a small number of elements/features will be very frequent, but that a large number of elements/features will be quite rare, which leads to exceedingly sparser matrices that give rise to convergence problems in regression models.
- Predictors are often somewhat or even quite collinear, which leads to unstable regression coefficients, which in turn can make it hard to assess the direction and the strength of an effect.

Given these issues, a growing number of researchers are exploring data with such characteristics using methods other than regression modeling; a particularly popular alternative showing up in more and more studies is the family of tree-based methods, including in particular classification and regression trees (CARTs), conditional inference trees, and random or conditional inference forests (see Tagliamonte and Baayen 2012; Dilts 2013; Hansen and Schneider 2013; Klavan et al. 2015; Szmrecsanyi et al. 2016; Rezaee and Golparvar 2017; Tomaschek et al. 2018; Hundt 2018, etc.). Tree-based methods function by repeatedly splitting data sets up into two parts such that the split leads to the best increase in terms of classification accuracy or in terms of some other statistical criterion (such as deviance or the Gini coefficient or others) when it comes to predicting the dependent variable. This family of approaches does not rely on estimations as parametric regression models do but can essentially be seen as a nonparametric alternative that is potentially much less affected by the kinds of data sparsity and collinearity that make regression models of observational data so difficult. In addition, there is a perception that the tree-based visualization format is more intuitively interpretable than the summary/coefficients table usually reported for regression models (see Baayen et al. 2013: 265 for such an assessment).[1] For

1 In addition, there is also a perception that classification trees can offer insights that regression models cannot; in particular, people believe that they can represent interactions much better than regressions can. However, the situation is more complex than that. It is true that trees can easily display, say, a four-way interaction or even higher order ones that a researcher would be unlikely to fit in a regression model (see Figure 2). At the same time, researchers usually do not have predictions at this high level of resolution and are therefore often not really interested in the interactions of this amount of complexity. When did one last see a prediction in a paper regarding a four-way interaction? I am not sure I ever did .... On the other hand, the kinds of interactions between predictors that researchers usually *are* interested in – two-way interactions, maybe three-way interactions – can be fitted and visualized perfectly well in regression models (which are also often better at visualizing effects of numeric predictors, see below). Be that as it may, one of the central points of this paper is to show how trees, and to a lesser extent forests, can fail at identifying even a very simple two-way interaction.

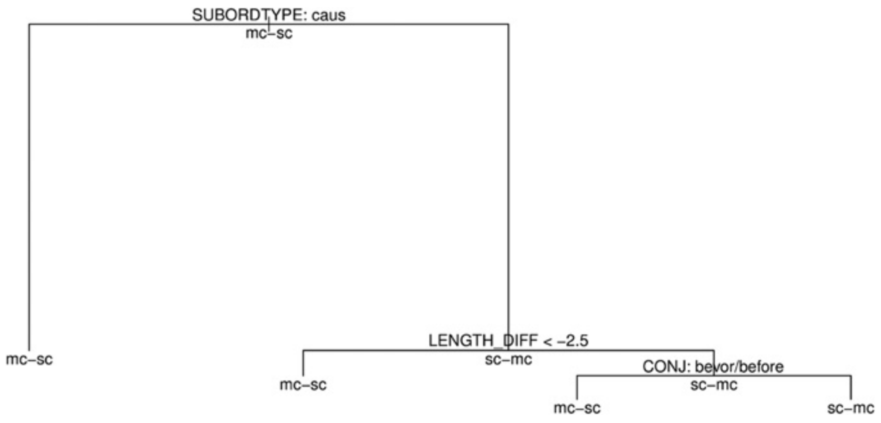

Figure 1: Classification tree "predicting" whether a main clause will precede a subordinate clause (*mc–sc*) or whether it will follow it (*sc–mc*).

instance, Figure 1 shows the results of an analysis of the corpus data on main-subordinate-clause ordering in the INTERSECT corpus discussed in Gries and Wulff (2012) and Gries (2013); the critical question was which of the two clause orders exemplified in (1) is preferred and which impact, if any, do the following predictors have on the ordering choices speakers make:

– CONJ: which conjunction heads the subordinate clause: *weil/because* versus *bevor/before versus als/when* versus *nachdem/after*;
– SUBORDTYPE: causal or temporal (as in (1)); and
– LENGTHDIFF: the length difference between the main and the subordinate clause (in words).

(1) a. I was shocked when the cat killed the mouse.
 b. When the cat killed the mouse, I was shocked.

The classification tree resulting from the data can be interpreted as follows: Starting from the top, if the subordinate clause type is *causal*, go left and "predict" *mc–sc*; if the subordinate clause type is *temporal*, then go right and check the length difference of the main and the subordinate clause: if that difference is < –2.5, go left and also "predict" *mc–sc*, otherwise go right and check whether the conjunction is *before*. If it is, "predict" *mc–sc* again, but if it is not, "predict" *sc–mc*.

This particular classification tree can actually be summarized more concisely, given its structure: One could just say "always 'predict' *mc–sc*, unless the length difference between main and subordinate clause is greater than –2.5 *and* the conjunctions are *after* or *when*." However, this tree actually already also suggests that summarizing such a tree in prose can actually be extremely nonintuitive/difficult. This is for several reasons: First, one needs to realize that every split in the tree potentially requires another *if*-clause – "if X is this, go left, if it isn't, go right" – which means that, with increasing depths of such trees, their prose summaries become quite painful to process. Figure 2 is an example of a conditional inference tree (predicting a four-level dependent variable) whose interpretation is in fact quite challenging, the seemingly nice plot notwithstanding: I invite the reader to try to describe the split happening at node 13 in a way that is less painful than this: "if the recipient is an animate NP up to five words long and if the patient is two or three words long and not concrete, then … but if the recipient is an animate NP up to five words long and if the patient is two or three words long and not concrete, then …" I am not sure this is as "intuitive" as trees are often claimed to be (see Szmrecsanyi et al. 2016 for other examples of trees that are quite complex in terms of how they would need to be summarized in prose).

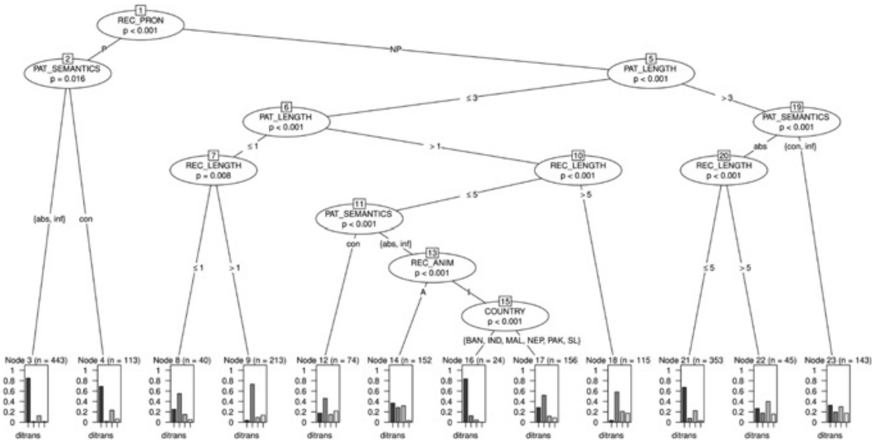

Figure 2: Conditional inference tree (Figure 1 of Bernaisch, Gries, & Mukherjee 2014).

Second, one also needs to realize that the binary-split nature of these methods can make the interpretation of the effects of numeric predictors quite cumbersome: Where a regression model would just return a significant slope for the effect of a numeric predictor, which can be easily summarized with a "the more *x*, the more/less *y*" sentence, a classification tree will often "represent" such a slope with multiple binary splits on the same variable. Consider the verb-particle construction alternation exemplified in (2), which is known to be quite strongly influenced by the length of the direct object (DO):

(2) a. The cat bit off the bird's head.
b. The cat bit the bird's head off.

Consider how the effect of this numeric predictor is represented in a classification tree such as the one shown in Figure 3, which uses a practice data set from Gries (2013) in which 200 examples of the alternation in (2) are annotated for the following predictors:

- MODE: whether the example is from spoken or written data;
- COMPLEXITY: how complex the DO is: *simple* versus *phrasally modified* versus *clausally modified*;
- DO_LENGTHSYLL: how long the DO is (in syllables);
- PP: whether the verb-particle construction is *followed by a directional PP* or *not*;
- DO_ANIMACY: whether the referent of the DO is *animate* or *inanimate*;
- DO_CONCRETENESS: whether the referent of the DO is *concrete* or *abstract*.

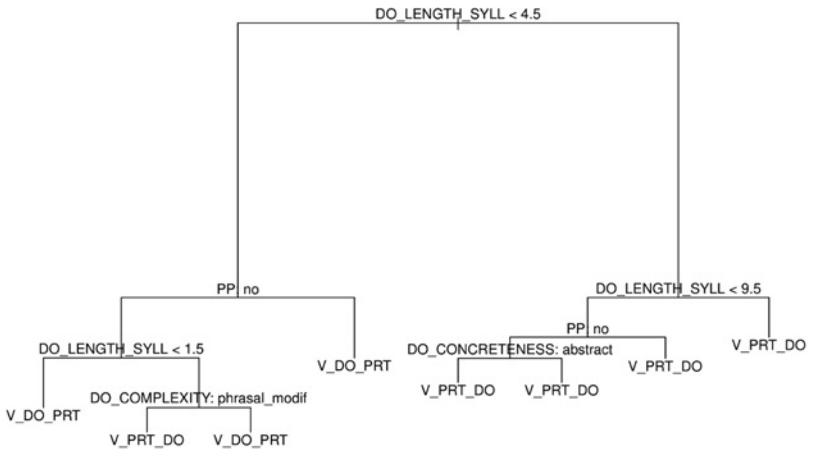

Figure 3: Classification tree "predicting" whether a particle will precede a direct object (V_PRT_DO as in (2)a) or whether it will follow it (V_DO_PRT as in (2)b).

The tree reveals the effect of DO_LENGTHSYLL in two splits: First, there is a split at the top depending on whether the DO is shorter than 4.5 syllables. But if it is and one goes down the left part of the tree, later one needs to also consider whether the DO is shorter than 1.5 syllables (i.e. whether the DO is one syllable long). And if the DO is not shorter than 4.5 syllables and you go down the right part of the tree, then later one needs to consider whether the DO is less than 9.5 syllables long. In other words, it can happen that that single number in a regression model – a slope of a numeric predictor – is recoverable from a tree only by piecing together three or even more splits in different locations in a classification tree.

Finally, it is worth mentioning that classification trees are not always particularly stable or robust: (1) even small changes in predictor values can produce big changes in the predictions, (2) even small changes to a data set can lead to drastic changes in the structure these kinds of algorithms find (Strobl et al. 2009) and represent, and (3) overfitting can be a problem, which makes it very important that trees are subjected to rigorous validation and pruning (see e.g. Kuhn and Johnson 2013: 182, Crawley 2013: Section 23.5, or James et al. 2013: 307–312), something that is often at least not reported on.

One recent attempt to improve tree-based approaches that will also be included in this discussion is that of conditional inference trees, a version of trees that uses a regression- and *p* value-based approach to identify the best splits in predictors (rather than the abovementioned criteria used in many regular tree-based approaches), which reduces the need for pruning and cross-validation (Hothorn et al. 2006b) and addresses the problem of many tree-based approaches that "variables offering more cut points are artificially preferred in variable selection, [... a bias that] is carried over to the Gini variable importance measure" (Boulesteix et al. 2015: 342).

Another recent development is that studies are now also more often using an extension of tree-based methods called random forests (Breiman 2001). Random forests add two layers of randomness to tree-based methods:

– randomness on the level of the *data points* because random forests involve fitting many decision trees on many different randomly sampled (with or without replacement) subsets of the original data;
– randomness on the level of *predictors* because at every split in every tree, only a randomly selected subset of the predictors is eligible to be chosen.

Random forests usually tend to overfit much less than individual CARTs or training/test validation approaches and can be straightforwardly evaluated given that random forest accuracies are *prediction*, not *classification*, accuracies. In other words, while classification trees, strictly speaking, only return classification accuracies, i.e. accuracies of classifying cases on which the tree was trained, random forests return predictions for cases that did not feature in the training process of each tree, but cases that were "out of the bag" of a tree that was fit, thus reducing overfitting and obviating the need for cross-validation.

However, compared to regression models and classification/conditional inference trees, random forests have the distinct disadvantage that interpreting them is hard: There is no single model from a single data set, or no single tree from a single data set, that can be straightforwardly plotted – there are, say, 500 different trees fit on 500 differently sampled data sets, with thousands of different splits in different locations in trees with differently sampled predictors available for splits .... Thus, the problem is that there is a method (random/conditional inference forests) that seems statistically superior to another (classification/conditional inference trees), but that is much harder to interpret/visualize. Two solutions are pursued in the literature:

- The random forest implementations that seem to be most widely used in (corpus) linguistics offer the functionality of computing *variable importance scores*, which quantify the size of the effect that a predictor has on the response; some version of these – permutation-based scores, conditional importance scores, and scaled or unscaled ones – are reported frequently. In addition, they offer the computation of *partial dependence scores*, which represent the direction of the effect that levels/values of the predictor have on a response; for reasons not entirely clear to me, these are reported much less often (not even by scholars using Rs randomForest package, where these scores can be generated very easily).
- Since the publications of Tagliamonte and Baayen (2012) or Baayen et al. (2013), researchers have also sometimes opted to interpret a random forest grown on a data set using a classification/conditional inference tree grown on the same data set. In other words, effect sizes might be reported in the form of variable importance scores resulting from the random forest analysis, whereas the direction of the effects might be reported from the single tree fit on the data as a whole (as exemplified above).

1.2 The goals of the present paper

In this paper, I want to discuss critically the field's increasing reliance on tree-based methods as well as the field's currently predominant ways of using and interpreting random forests. To that end, I will first showcase a kind of data set that multiple tree-based approaches turn out to be very bad at handling in the sense that they fail to identify the correct predictors–response relation(s) in the data, which can lead to (1) suboptimal classification accuracies and (2) non-parsimonious trees (i.e. suboptimal variable importance scores); this data set and its analysis using the following R functions: `tree::tree` and `rpart::rpart` for classification trees and `party::ctree` as well as `partykit::ctree` for conditional inference trees (although the latter is merely a "reimplementation" of the former, see <http://jmlr.org/papers/v16/hothorn15a.html>) are discussed in Section 2; in all cases, unless stated otherwise, I am always using the default settings of these and other functions, but the issue with the data and, thus, logic of the paper are not affected by the kinds of tweaks that, if any at all, are sometimes made. In other words, maybe one can find a very specific set of settings that might ameliorate the problem *a posteriori*, but in all normal applications, changing the settings does not matter much for the specific point I am trying to make.

Second and based on that first part, I will discuss how the practice of summarizing random forests on the basis of single trees can be, minimally, risky or, maximally, even flawed and how the kinds of data that are problematic for tree-based approaches can be studied more safely; these issues will be the focus of Section 3. Code for all the analyses in Sections 2 and 3 will be made available on my website. Section 4 concludes.

2 Data sets that trees cannot handle (very well)

2.1 A small artificial data set

Let me first introduce a data set that has a structure that we will find is problematic for the tree-based approaches; this data set is represented in a summary frequency table in Table 1.

There are two crucial things to notice about this data set. The first is that the three predictors differ in monofactorial predictive power:

Table 1: Data set 1_{small}.

Predictor 1	Predictor 2	Predictor 3	Binary response variable	
			Level: *x*	**Level: *y***
a	e	m	5	0
a	e	n	0	3
a	f	m	0	0
a	f	n	2	0
b	e	m	1	0
b	e	n	0	1
b	f	m	0	6
b	f	n	2	0

- P1s (for Predictor 1 from the small data set) leads to 70% accuracy: when P1s is *a*, the Response is *x* 7 out of 10 times, when P1s is *b*, the Response is *y* 7 out of 10 times.
- P2s (for Predictor 2 from the small data set) leads to 60% accuracy: when P2s is *e*, the Response is *x* 6 out of 10 times, when P2s is *f*, the Response is *y* 6 out of 10 times.
- P3s (for Predictor 3 from the small data set) leads to only 50% accuracy: when P3s is *m*, the Response is *x* 6 out of 12 times, when P3s is *n*, the Response is *y* 4 out of 8 times.

The second thing to notice is that the two weaker predictors, P2s and P3s, together yield perfect 100% accuracy:
- when P2s is *e* and P3s is *m*, the Response is *x* all 6 times;
- when P2s is *e* and P3s is *n*, the Response is *y* all 4 times;
- when P2s is *f* and P3s is *m*, the Response is *y* all 6 times;
- when P2s is *f* and P3s is *n*, the Response is *x* all 4 times.

This data set exemplifies something that is at least relatable to the so-called XOR problem,

> a situation where two variables show no main effect [not true of P2s. which has a main effect] but a perfect interaction. In this case, because of the lack of a marginally detectable main effect, none of the variables may be selected in the first split of a classification tree, and the interaction may never be discovered. (Strobl et al. 2009: 341)

How do different tree-based methods handle these data? In the following subsections, I will discuss the applications of two kinds of classification trees and two kinds of conditional inference trees to these data; every tree application will be evaluated on three dimensions: (1) the classification accuracy that the tree yields, (2) the variables' importance, and (3) the effect that each predictor has in each tree.

One final comment regarding these data: The use of the above data set is not to imply that a data set like this is typical for corpus-linguistic data in general or for tree-based analyses of such data in particular. Of course, corpus data are usually – hopefully – a bit larger than the above data (and see Section 2.2 below for a larger sample size) and they do not usually exhibit the kind of complete separation shown above. However, the Zipfian distribution that corpus data often exhibit makes it quite likely that some categorical predictors have highly frequent levels whose association to the response variable may overpower other (combinations of) predictors. Also and as even one reviewer commented, this data set "seems to be the best (because simplest) illustration of the issue at hand" because it allows me to show that tree-based approaches may even fail at detecting the *perfectly predictive* effect of P2s and P3s in the data; the tree-based methods discussed below would not miraculously fare better if the data were not perfectly predictive (as that reviewer actually showed on the basis of more probabilistic/less deterministic data).

2.1.1 How `tree::tree` handles this small data set

The first tree is based on `tree::tree` in R, i.e. the function tree from the package tree (Ripley 2018). The default settings of this algorithm, which I am using here as well (given how most tree-based studies do not report hyperparameter settings at all or none that differ from the defaults), result in the tree shown in Figure 4. The tree splits first on P1s, because that is the most powerful predictor monofactorially as discussed above. If P1s is *a*, then the tree next splits on P3s, which means that the tree's terminal nodes distinguish three scenarios:

- when P1s is *b*, the tree predicts *y*, which predicts seven out of ten cases correctly;
- when P1s is *a* and P3s is *m*, the tree predicts *x*, which predicts all five cases correctly;
- when P1s is *a* and P3s is *n*, the tree predicts *y*, which predicts three of five cases correctly.

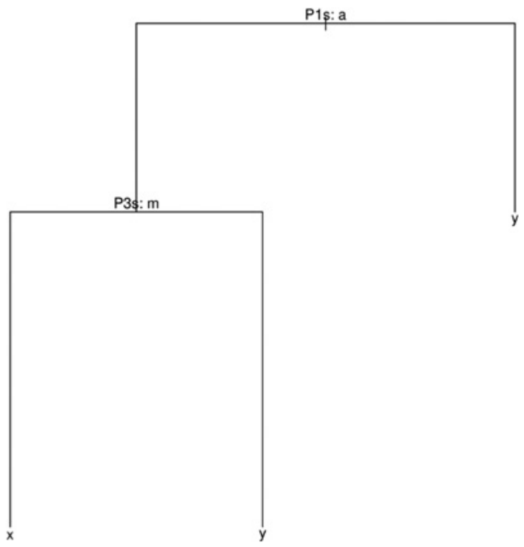

Figure 4: The result of `tree::tree` when applied to data set 1_{small}.

Thus, the accuracy of this tree is $(7+5+3)/20=75\%$ and is achieved by a combination of P1s and P3s. This is of course a disappointing result because we know that 100% accuracy is easily attainable, but by *not* including P1s at all, but by including P2s and P3s in what in regression modeling would be the interaction P2s:P3s. The problem is that the tree algorithm checks all three predictors, finds that P1s is monofactorially most powerful, and then runs with it, and the fact that P2s:P3s can actually solve the classification task perfectly is never recovered because the algorithm never "goes back" to check if some other first choice might work better.

2.1.2 How `rpart::rpart` handles this small data set

The second tree uses the function `rpart` from the package `rpart` (Therneau and Atkinson 2018) and is shown in Figure 5. This tree splits first and *only* on P1s, because that is the most powerful predictor monofactorially as discussed above. If P1s is *a*, the tree predicts *x* (mispredicting three out of ten cases); if P1s is *b*, the tree predicts *y* (mispredicting another three out of ten cases). Thus, the accuracy

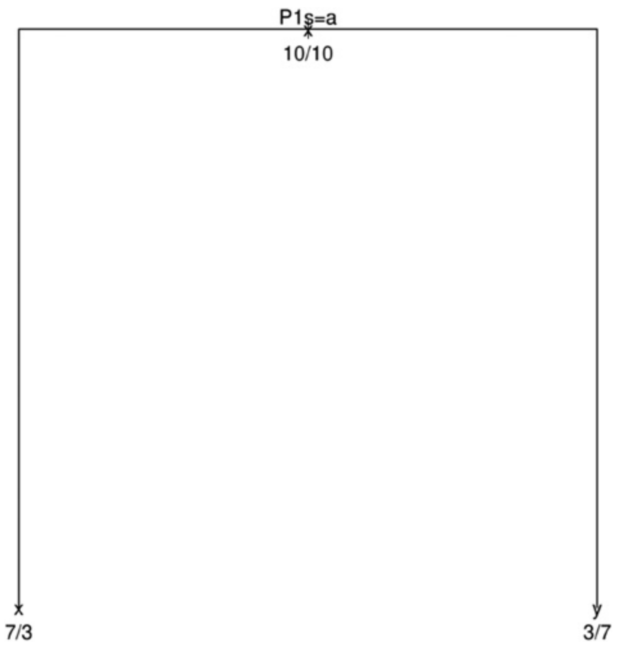

Figure 5: The result of `rpart::rpart` when applied to data set 1_{small}.

of this tree is an even worse 70%. Again, the problem is that the tree algorithm checks all three predictors, finds that P1s is most powerful, and runs with it, and then the `rpart::rpart` algorithm does not even do any other split, which means it attributes too little variable importance – none in fact – to both P2s and P3s.

2.1.3 How `party::ctree` handles this small data set

Let us finally look at a conditional inference tree from the function ctree of the package party (Hothorn et al. 2006b). The result is shown in Figure 6, and this result is the worst of all because it is not even a tree. Instead, the result is a stacked bar plot that just shows the frequencies of the two levels of the response: of the two levels of the `ctree` does not use any predictor at all, which means that its

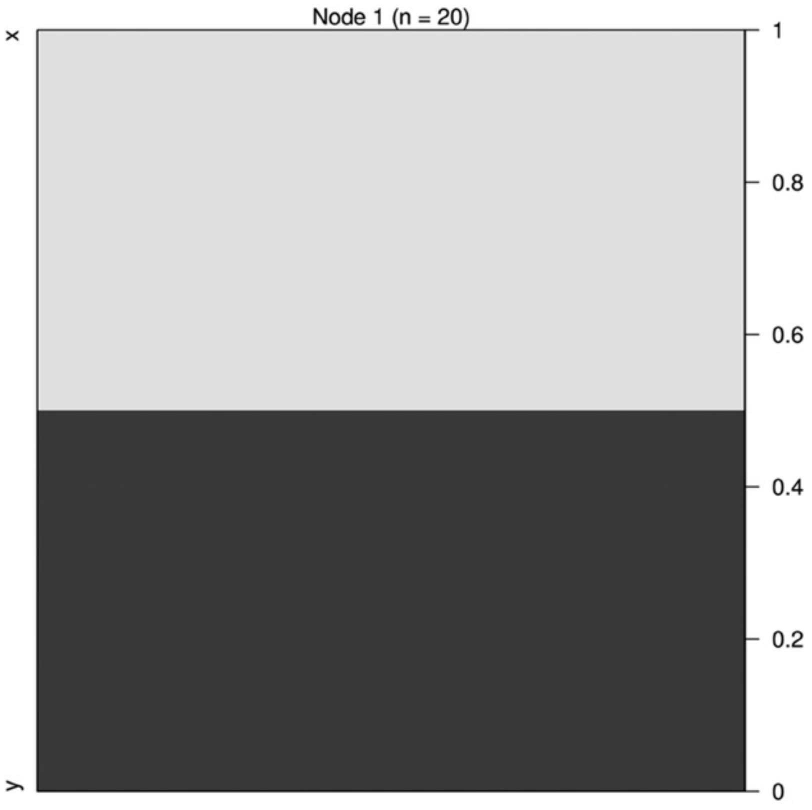

Figure 6: The result of `party::ctree` when applied to data set 1_{small}.

accuracy is at baseline/chance level, i.e. here 50% and all variables have importance scores of 0. The same results are obtained with `partykit::ctree` (Hothorn and Zeileis 2015).

2.1.4 Interim summary

In sum, in terms of accuracy, none of the approaches scores a value higher than 75% although there is a very simple interaction-like structure in the data that should result in 100% accuracy. In terms of variable importance, no tree recognizes that P2s and P3s are important when combined, whereas P1s is not: Two approaches exaggerate the role of P1s and the third doesn't recognize any predictor as important; with these results, no approach would produce proper partial dependence scores for P2s and P3s: Even `tree::tree`, the only method that attributes at least *some* importance to P3s, does not see that P3s interacts with, so to speak, P2s. In other words, none of the trees succeeds at finding the right structure in even as simple a data set as this and this is because, as Strobl et al. (2009: 333) point out,

> the split selection process in regular classification trees is only locally optimal in each node: A variable and cutpoint are chosen with respect to the impurity reduction they can achieve in a given node defined by all previous splits, but regardless of all splits yet to come.

2.2 The large version of the artificial data set

Given these results, a first obvious objection to the above would be that part of the problem might be the ridiculously small sample size, which should affect especially the conditional inference tree, which uses *p*-values as a splitting criterion and could be expected to suffer from the small sample size. Let us therefore increase data set 1_{small} by a factor of 10 (and rename the predictors to P1l [for *large*], etc.) and see whether that improves the results. As we will see, the new sample size leads to considerable changes.

2.2.1 How `tree::tree` handles this larger data set

The algorithm from `tree::tree` now returns a classification accuracy of 100%: every single case is identified correctly. However, as Figure 7 shows, the tree is still not ideal because the tree still first splits on P1l. In other words, the tree is not ideal in terms of parsimony: it uses a predictor we *know* is not needed to get to

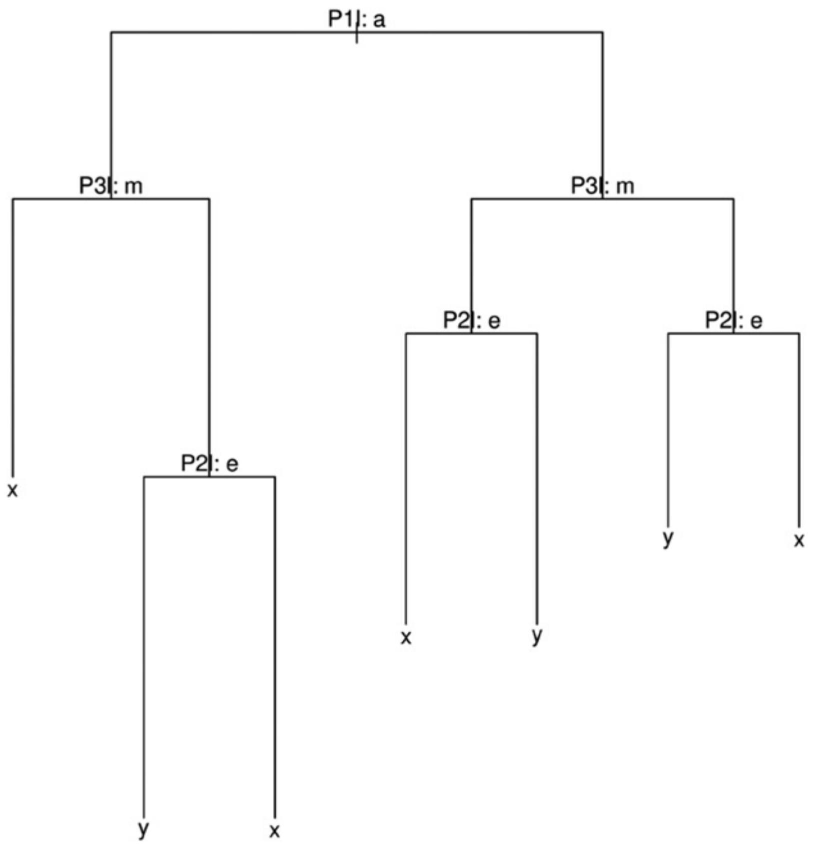

Figure 7: The result of `tree::tree` when applied to data set 1_{large}.

100% accuracy, which of course also means it assigns a variable importance value to P1l that a researcher would want it to be. In terms of variable effects, the tree recognizes the interaction of P2l:P3l, which is how it arrives at 100% accuracy in the first place. In other words and in the parlance of regression modeling, the tree suggests the data should be explained with the three-way interaction P1l:P2l:P3l when we know that the two-way interaction P2l:P3l is sufficient.

A somewhat tree-savvy reader might now inquire whether the result of `tree::tree` will improve in terms of parsimony if the tree is pruned, a procedure that is routinely employed to avoid overfitting. However, as the companion file on my website at <http://www.stgries.info> shows, this is not the case: If the tree is pruned (either on the basis of misclassifications or on the basis of deviance), its accuracy is reduced from 100% to 85%, and the tree *still* splits on P1l first and, thus, *still* overestimates P1l's importance or still implies the importance of the three-way interaction P1l:P2l:P3l.

2.2.2 How `rpart::rpart` handles this larger data set

The increase of the sample size also changes the results of `rpart::rpart`, and the resulting tree shown in Figure 8 is actually the same as that of `tree::tree` in Figure 7, which means that all the above evaluative comments apply here as well, especially since pruning the tree also leads to the same results as above: a reduction of accuracy from 100% to 85% while still leaving P1l in the tree.

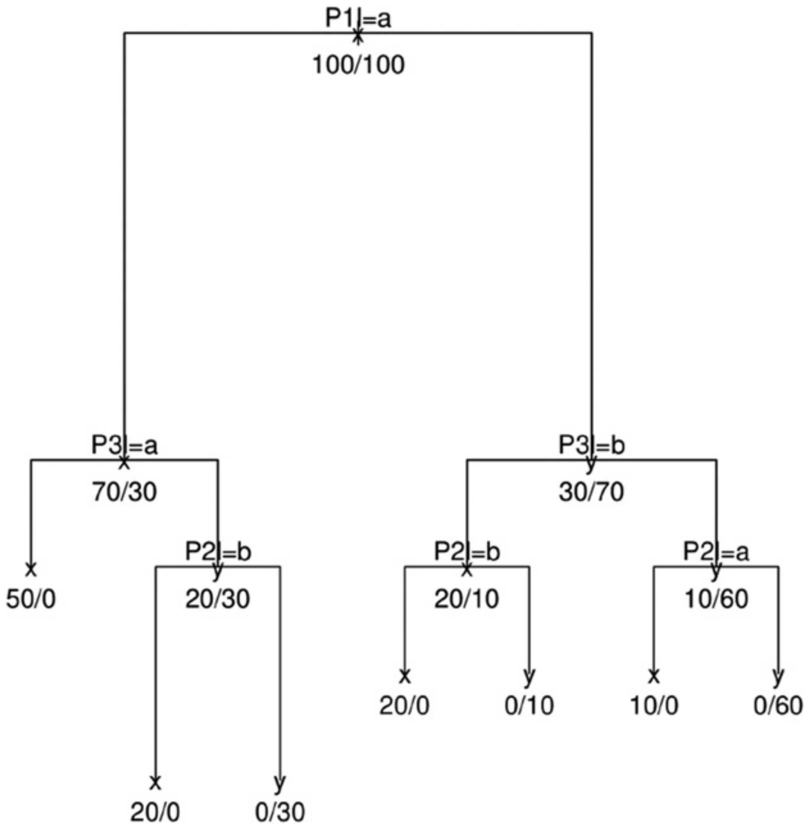

Figure 8: The result of `rpart::rpart` when applied to data set 1_{large}.

2.2.3 How `party::ctree` handles this larger data set

What about the conditional inference tree, whose performance should benefit considerably? Indeed, this conditional inference tree (and the corresponding tree generated with `partykit::ctree`) now leads to the same results as the other two methods, as is shown in Figure 9.

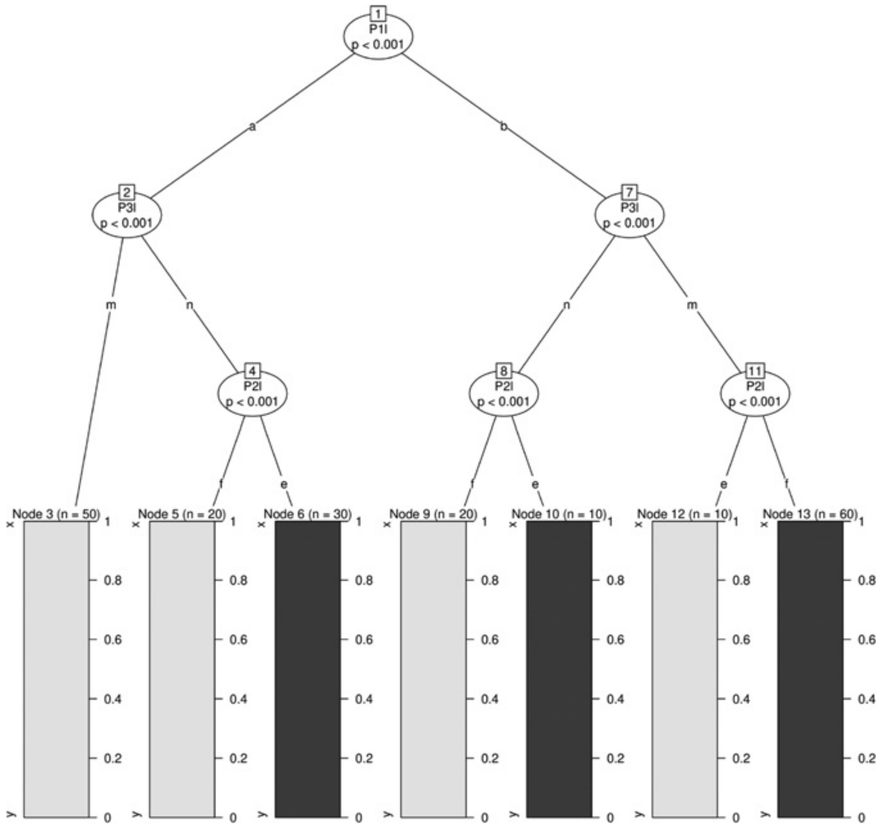

Figure 9: The result of `party::ctree` when applied to data set 1_{large}.

2.2.4 Interim summary

The more realistic sample size of data set 1_{large} has changed the picture considerably: All approaches now achieve 100% accuracy, but still no approach recognizes that it is the interaction of P2l:P3l alone that would be sufficient;

none of the trees is parsimonious. This is already an interesting finding given how widespread the consensus among corpus linguists is that tree-based approaches are good at detecting interactions: in this case, all approaches return a three-way interaction when a two-way interaction is all that would be required/desired.

The issue of variable effects is worthy of specific mention: If we were to compute variable importance scores on the trees, then these variable importance scores for P1l, P2l, and P3l do not alert the analyst to the fact that it is the "interaction" of P2l and P3l – not P2l or P3l in isolation – that does *all* the work. In other words, partial dependence scores would not provide the desired results in any of the above applications – what we would ideally like to see is (1) a variable importance score that is very small for P1l and (2) some indications that P2l and P3l on their own do not do much, but that, together, they do a lot.

2.3 Random/Conditional inference forests to the rescue?

On the whole, the picture that has emerged so far is somewhat sobering because trees on both the small and the large data sets never returned an optimal tree, *optimal* in terms of accuracy, parsimony, *and* effect interpretations simply because one strong predictor chosen for the first split may overpower everything else, something which can of course very easily happen in Zipfian-distributed corpus-linguistic data. Is there any way in which this result can be improved?

One approach would seem to be to try and tweak hyperparameters of fitting trees, such as (1) the minimum decreases in deviance that define when trees stop splitting, (2) the minimum sample sizes per node, or (3) the tree depth. In the current scenario, however, they will not change that the tree-building algorithms will still go with local importance and split on P1s/P1l. Thus, a more powerful improvement might be the use of the extension of classification/conditional inference trees already discussed in Section 1.1, namely, random forests. Specifically, one might expect that the two layers of randomness introduced in random forests would help in the present case because, among the, say, 500 trees,

– some could contain random samples of the data points in which P1s/P1l is not as dominant as it is in the data set as a whole;
– some could be trees where P1s/P1l was not available for the first split, which means the algorithm could only choose either P2s/P2l or P3s/P3l, which in turn means that if, at the next splits, P3s/P3l or P2s/P2l were available, respectively, those trees would result in 100% accuracy and parsimoniously so it also means these trees would assess P1s/P1l's importance like a human analyst might prefer it (see Strobl et al. 2009: 333).

Table 2: Variable importance scores from randomForest::randomForest on data set 1_{small} and data set 1_{large} (computed with randomForest::importance).

Variable importance for data set 1_{small}		Variable importance for data set 1_{large}
1.651851	Predictor 1	12.93793
2.950552	Predictor 2	20.09449
3.618863	Predictor 3	51.40719

In other words, the fact that not all predictors are always available for splits addresses the problem that “a classification tree makes its splits based on local best performance” (Baayen et al. 2013: 265), as already mentioned above.

What happens if we apply `randomForest::randomForest` (Liaw and Wiener 2002) to both data set 1_{small} and data set 1_{large} (setting the hyperparamater of mtry [how many predictors are considered at each split?] to 2 so that there is at least some choice at each split and using the default of ntree [how many trees are grown?]), first, the accuracies are improving over the trees: The random forest from data set 1_{small} scores an accuracy of 90%, whereas the forest from data set 1_{large} scores 100% (i.e. just like the trees). However, as Table 2 shows, there *is* a marked improvement in terms of the variable importance scores: For both data set 1_{small} and data set 1_{large}, the random forest “recognizes” that P1, which we know to be actually completely unnecessary, is least important. (The results from `ranger::ranger` [Wright and Ziegler 2017] are conceptually the same.)

What about the interpretation of the effects, however? While using random forests has not improved accuracy much but only variable importance scores, the interpretation of the random forests using partial dependence scores is still somewhat problematic: Figure 10 shows the partial dependence scores the function `randomForest::partialPlot` returns for each data set and, given our knowledge of the data, we immediately recognize two related problems.

First, the values of the partial dependence scores in the small data set are nearly perfectly aligned with the monofactorial accuracy percentages each predictor can score: P1s’s scores deviate from 0 most (corresponding to the fact that it scores 70% accuracy), followed by P2s and then P3s. In other words, in this case, the random forest’s partial dependence scores just replicate the simple observed percentages attainable from cross-tabulation. For the large data set, the situation is better, but the difference between the (totally irrelevant) predictor P1l and the predictors P2l and P3l (highly relevant in an interaction) is way smaller than we would like it to be. Second and relatedly, this of course also means that we do not get a score for the interaction of P2s:P3s/P2l:P3l.

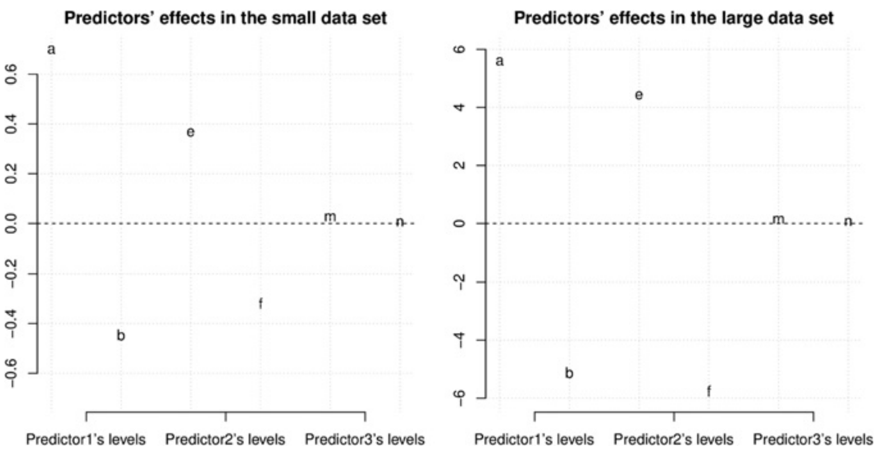

Figure 10: Partial dependence scores for the random forests on data set 1_{small} and data set 1_{large}.

What if we generate a random forest out of conditional inference trees instead (using `party::cforest`, see Hothorn et al. 2006a; Strobl et al. 2007, Strobl et al. 2008), ignoring for the moment (like most scholars) that the corresponding help page for `party::cforest` states "Ensembles of conditional inference trees have not yet been extensively tested, so this routine is meant for the expert user only and its current state is rather experimental"? Using the same number of variables available for each split as with `randomForest::randomForest`, the accuracies returned by `party::cforest` for data set 1_{small} and data set 1_{large} are similar to those of `randomForest::randomForest`: 70% and 100%, respectively, and Table 3 contains the variable importance scores obtained from those forests; at least for the larger data set, `party::cforest` and `partykit::cforest` recognize the lesser importance of P1 and particular so with the conditional variable importance scores that really downgrade P1 as one would want it.

Table 3: Variable importance scores from party::cforest on data set 1_{small} and data set 1_{large}.

Variable importance for data set 1_{small} standard/ conditional		**Variable importance for data set 1_{large} standard/ conditional**
0.05353/0.04798	Predictor 1	0.14388/0.01227
−0.00838/−0.01015	Predictor 2	0.22518/0.20476
−0.01155/−0.01156	Predictor 3	0.3229/0.31795

In sum, the switch toward random forests helped, but only a bit: The accuracy of the forests is good but not all variable importance scores tell the story we would like to be told given the structure of the data, and the partial dependence scores do not reveal the interaction effect we would be most interested in. However, this also leads to the next problem: If forests are on the whole doing better than trees on one and the same data set, i.e. the two methods yield very different results in terms of quality, then how can one possibly use the latter to summarize/interpret the former?

3 The practice of summarizing a forest with a tree on all the data is problematic

As just mentioned, the above discussion already indicates why summarizing a random forest with a tree grown on all the data is risky or worse. First, this practice seems problematic already on a very general level: How could a single tree that was fit on all the data (no sampling of cases) and with all predictors all the time (no sampling of predictors) possibly be great at summarizing a forest of hundreds or thousands of different trees based on that very sampling of cases and predictors?

Second and more concretely here, it is problematic on empirical grounds: We saw above that the random forest results are not necessarily reflected well by any of the trees. Specifically, if one uses any of the trees grown on data set 1 to "visualize the random forest" grown on data set 1, then one runs into the problem that the trees either discovered no structure at all or considered P1 to be (one of) the most important predictors, while at the same time, P1 is the predictor that the random forests return as less or least important; in other words, how can Figures 4–9 possibly be seen as summarizing the results underlying Tables 2 and 3? And, of course, there is still the problem that none of the methods so far reflected the right predictor structure anyway, namely, the perfectly predictive power of P2:P3? Yes, (especially the conditional) variable importance measures of the forests indicated that P2 and P3 are more important than P1, but they did not indicate that it is the interaction of these predictors that is important rather than their main effects.

Not only is this ironic given how many people consider tree-based approaches as good at finding/representing interactions, but there is also a much more important consequence: The above results also point to an issue that seems not to be discussed in linguistic applications of random forests, namely, the question of how good forests and their variable importance measures are at *capturing* interactions and/or *detecting* interactions, where *capturing* refers to a random forest identifying a variable that "contributes to the classification with an interaction effect" and where *detecting* refers to a random forest identifying "the interaction effect per se and the predictor variables interacting with each other" (Wright et al. 2016: 1). In fact, corpus data might be a perfect storm kind of data for forests:

– Wright et al. (2016: 7) demonstrate that "random forests are capable of *capturing* [...] interactions" (my emphasis) but that both single and pairwise importance measures "are unable to *detect* interactions in the presence of marginal effects" (my emphasis), and of course in corpus-linguistic data, we often have both strong marginal effects and interactions, i.e. the combination that makes variable importance measures bad at detecting interactions.
– Boulesteix et al. (2015: 341) argue that "strong association between predictor variables may in some cases hinder the detection of interaction effects, especially in data sets of moderate size," and of course in corpus data, we often have highly associated/correlated predictors and only moderate data sizes (compared to the kind of data that much work on random forests is done on, i.e. genetics data with hundreds of thousands of data points).

Thus and exactly as demonstrated by our above data,

> RF methodologies are commonly claimed, often in rather vague terms, to be able to handle interactions [...], although, by construction, the predictor defining the first split of a tree is selected as the one with the strongest main effect on the response variable. (Boulesteix et al. 2015: 344)

These things raise important questions when it comes to the interpretation of probably most (corpus-)linguistic studies involving random forests: It is very well possible that previous studies using random forests,

– interpreted variables with high importance scores as main effects although the high importance score was due to the variable participating in an interaction (which the study then did not discuss);
– interpreted variables with high importance scores as being important although the high importance score was due to the variable being correlated with one or more other predictors (which was not discussed because scholars think random forests are less sensitive to collinearity and, therefore, might not even explore it in their data);
– interpreted random forests with a single tree and thus (completely?) misrepresent the findings of the forest (as discussed above).

All these of course also raise the question of what to do instead. In what follows, I offer a few suggestions that I would encourage researchers to explore.[2]

3.1 Improving trees' and forests' performance: interaction variables as predictors

The first possibility to make the tree-based approaches – both "simple" trees and random forests – better at recognizing the predictive structure in the data is to force the trees to consider the combined effect of P2 and P3 by explicitly including the combination, the interaction, of these two predictors or, more generally, all predictors in whose interactions one might be interested in. This approach is similar to the logic of Forina et al. (2009), who "augment [...] the prediction data with addition of complex combinations of the data" (Kuhn and Johnson 2013: 48). For this data set, it just means we, as in Kuhn and Johnson (2013: 57f.), create three new predictors:

- P1:P2, the interaction of P1 and P2; i.e. this predictor will have the levels *ae, af, be*, and *bf*;

2 A reviewer pointed out correctly that it is sometimes recommended to use a tree to delimit the space of possible predictors for a subsequent regression model. For instance, one might run a tree on the data discussed so far, see that somewhere in the tree there seems to be an interaction of P2 and P3, and therefore include that interaction in a regression equation. However, this is tricky if not outright problematic in two ways. First, a tree will usually not give unambiguous clues as to which interactions to include. Recall the complexity of Figures 2 and 3 again: In trees in general, it would not be obvious what interactions to include: Do we use all two-way interactions, all three-way interactions, all *n*-way interactions represented by all splits? Do we let the lengths of the vertical lines between split delimit this interaction set? And let's not forget the fact that Figures 7–9 amount to suggesting the three-way interaction of P1l:P2l:P3l is relevant/required.

Second, from a very strict hypothesis-testing perspective, this procedure could probably be called cheating. Imagine, one has a scenario with many predictors A–Z (which could be the main effects and/or interactions), then one uses trees to figure out that A, B, E, G, and M are important, and *then* one does a regression with those and, miraculously, finds that the predictors that are significant are, wait for it, A, B, E, G, and M. I know that this kind of recommendation – tree first, then regression modeling – is out there, but this recommendation is not straightforwardly compatible with much of the logic that significance testing is (supposed to be) about; as Strobl et al. (2009: 341) correctly point out, "variable selection (with, say, random forests) should not be conducted before applying another statistical method on the same learning data." Third and relatedly, most applications of trees that I see are *not* used as a preparation for regression modeling; they are used *in place of* regression modeling, often by, no disrespect, scholars who have understood we need multifactorial methods, but who still lack the knowledge to handle (mixed effects) regression modeling and all that entails.

- P1:P3, the interaction of P1 and P3; i.e. this predictor will have the levels *am, an, bm*, and *bn*;
- P2:P3, the interaction of P2 and P3; i.e. this predictor will have the levels *em, en, fm*, and *fn*.

See Strobl et al. (2009: 341) for what might be the same suggestion, and then we make these new variables part of the list of predictors given to the tree-building functions, i.e. the formulae change from (3a) to (3b):

(3) a. Response ~ P1 + P2 + P3
  b. Response ~ P1 + P2 + P3 + P1:P2 + P1:P3 + P2:P3

What happens if we refit the tree-based approaches on both the small and the large versions of the data? Now, everything is perfect:
- Accuracy: Every single tree scores 100%.
- Variable importance: Every single tree chooses P2:P3 as its only predictor, thus, each recognizes that P2:P3 is the only predictor that is important (and thus the trees are all perfectly parsimonious: they contain all and only all important predictors).
- Variable interpretation: Every tree reflects perfectly that *em* and *fn* predict *x* and that *en* and *fm* predict *y*.

What about the random forests (from both `randomForest::randomForest` and `party::cforest`)? Again, the results are much improved:
- Accuracy: all but one forests score 100%.
- Variable importance: every single forest chooses P2:P3 as the by far most important predictor, but it is worth noting that, because of the sampling, all other predictors' variable importance scores are also not 0.[3]
- Variable interpretation: The partial dependence scores (computed on the forests produced by `randomForest::randomForest`), for instance, reflect that *em* and *fn* predict *x* and that *en* and *fm* predict *y*.

3 It is useful to point out that, when interaction predictors are included in conditional inference forests, the conditional variable importance scores cannot always be computed unproblematically anymore (because the "conditioning process" breaks down given that the interaction is perfectly determined by the main effects). My recommendation is to either stick with the standard ones or rely instead on the relatively recently implemented *AUC*-based (Area under the Curve) variable importance scores (Janitza et al. 2013), which are especially useful for dependent variables exhibiting the class imbalance problem (see the companion file).

An alternative pointed out to me by one reviewer is the functionality offered by the package `randomForestSRC` (Ishwaran and Kogalur 2019). If applied to a random forest using the function `randomForestSRC::rfsrc` but without interaction predictors fitted on the large data set, then `randomForestSRC::find.interactions` returns an output table that strongly encourages the analyst to consider also P2l:P3l.

A similarly interesting alternative could be `edarf:partial_dependence` (Jones and Linder 2017), which allows the user to specify a set of variables (e.g. P2l and P3l) whose combined predictive power is then computed so it can be compared to that of other sets; in the present case, specifying the set {P2l, P3l} returns a result that indicates the extremely high predictive power of the interaction of the two predictors.

A final potentially interesting method could be reinforcement learning trees, an improvement over random forests whose main characteristics sound exactly like what would be needed to address the problems discussed above:

> first, […] choose variable(s) for each split which will bring the largest return from future branching splits rather than only focusing on the immediate consequences of the split via marginal effects. Such a splitting mechanism can break any hidden structure and avoid inconsistency by forcing splits on strong variables even if they do not show any marginal effect; second, progressively muting noise variables as we go deeper down a tree so that even as the sample size decreases rapidly towards a terminal node, the strong variable(s) can still be properly identified from the reduced space; third, the proposed method enables linear combination splitting rules at very little extra computational cost. (Zhu et al. 2015: 1771, see `RLT::RLT`)

In sum, tree-based approaches *can* in fact be much less good at (1) being parsimonious and at (2) detecting interactions than is commonly assumed; this is true especially if interactions are not forced into a set of predictors explicitly and/or not explored once a first forest has been grown. However, once either of these options is pursued, both trees and random forests as used here can nearly always recover the structure in the data perfectly and parsimoniously.

3.2 Representing random forests: representative trees

Let us now turn to the question of how an existing random forest can be visualized better. We have seen that visualizing a random forest with a single tree fit on all the data can be highly misleading. However, a better option is available, namely, what is called a *representative tree*. This approach is implemented in Dasgupta (2014) and is based on exploring all the trees that constitute a random forest created with `randomForest::randomForest` and find one or more trees that are most representative of the forest. This functionality is available from `reprtree::Reprtree` and if that method/function is applied to, for instance, a random forest grown on data set 1_{large} *even without explicit interaction predictors*, it returns the representative tree shown in Figure 11, which visualizes precisely that interaction. While this approach of representative trees may of course not always succeed this spectacularly (different random number seeds lead to different results, which means fitting multiple forests with different random number seeds can be instructive, see Strobl et al. 2009: 343), it is theoretically much better motivated than simply fitting a single tree on all the data and, here at least, it is empirically motivated in how it recovers the structure we know characterizes the data best.

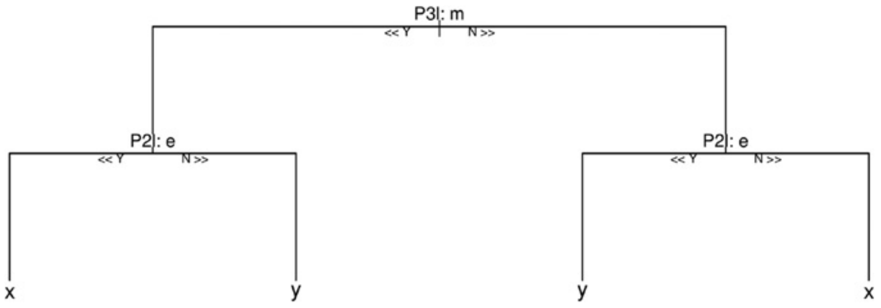

Figure 11: A representative tree for the random forest on data set 1_{large}.

3.3 Representing random forests: effects (plots)

Another possibility of representing some of the structure in a random forest – with or without interaction predictors – involves applying a logic/tool that is widely used in regression modeling contexts, namely, that of effects plots (Fox 2003). In a regression modeling context, the default effects plot for the following:

- A categorical predictor *x* represents the predicted values of the response variable for each level of *x*.
- A numeric predictor *x* represents predicted values of the response variable for a representative sequence of values from the range of *x*,

while, crucially, "values of other predictors [i.e. all those not currently being computed/visualized] are fixed at typical values: for example, a covariate could be fixed at its mean or median, a factor at its proportional distribution in the data" (Fox 2003: 1). This approach leads to nicely interpretable results for both main effects and interaction predictors that are computed while controlling for all other effects in a model. In the present context, this means one would

- generate a data frame that contains all predictors in the columns and all theoretically possible combinations of all predictor levels in the rows (called all.possible.combs in the code file for this paper);
- add to that data frame a column that provides the frequencies ff_{1-i} of each of these theoretically possible combinations of all predictor levels in the actual data (`all.possible.combs$COMBOFREQINREALDATA`);
- add to that data frame a column that contains the predicted probabilities of the outcome the regression/classifier tries to predict (`all.possible.combs $PREDPROB`);
- for each level l of a predictor of interest x, gather the predicted probabilities of the outcome pp_{1-i} of all predictor combinations that involve *x:l* (see the use of subset in the code file);
- the prediction for level l of predictor x is then computed as the weighted mean of pp_{1-i}, weighted by ff_{1-i} (see the use of weighted.mean in the code file).

For example, if one fitted a forest modeling the response variable in the large data set as a function of P1l, P2l, and P3l, this means that the predicted effect of P1l:*a* would be the mean of the predicted probabilities for the four combinations listed below weighted by their frequencies:

- combination P1l:*a* & P2l:*e* & P3l:*m* (with the frequency/weight of 50);
- combination P1l:*a* & P2l:*e* & P3l:*n* (with the frequency/weight of 30);
- combination P1l:*a* & P2l:*f* & P3l:*m* (with the frequency/weight of 0); and
- combination P1l:*a* & P2l:*f* & P3l:*n* (with the frequency/weight of 20).

This seems like an appealing approach because it represents a predictor's effect in a random forest in a way that is not only well known from regression modeling but also controls for all other predictors in a way that goes beyond what observed frequencies/percentages can provide. Some readers might recognize that this approach is similar to partial dependence scores provided by `randomForest::partialPlot`, but the above discussion is still relevant for two reasons: (1) If I understand the documentation for `randomForest::partialPlot` correctly (see the documentation of `partialPlot` and Molnar 2018: Section 5.1), that function computes only an *unweighted* mean of the predicted probabilities/logits (and the results from `pdp::partial` are the same), whereas the proposal above involves computing a *weighted* mean, i.e. one that is weighted by the frequency distribution of the actual data. This could be superior since it includes more information about our data (a logic that everyone using `effects::effect` seems to implicitly acknowledge). (2) As far as I can tell, the maybe more widely used packages `party` and `partykit` do not provide functions to compute partial dependence scores, which is why readers using those packages could use the logic discussed above to report the results of their conditional inference forests in a way that is widely used in regression modeling contexts.

3.4 Representing random forests: global surrogate models

One final and more preliminary suggestion for the interpretation of random forests involves the notion of global surrogate models (GSMs). They are based on the notion discussed by James et al. (2013: Section 2.1.3) that models and/or classifiers differ in terms of their power/flexibility to find patterns in data on the one hand and interpretability on the other; according to Kuhn and Johnson (2013: 50), there is an inverse relationship between the two. For instance, most regression models score high on interpretability but not quite as high on power/flexibility as other methods; by contrast, support vector machines, neural networks, and many new deep-learning methods are *very* powerful/flexible (given enough training data), but not easy to interpret. The GSMs are models of the easy-to-understand type that are fit on output from models of the hard-to-understand type so that the former can help understand the output of the latter.

In the present case, we could choose either one of two options:

- fit a binary logistic regression model on the categorical predictions of a random forest;
- fit a linear model on the predicted probabilities of Response: *y* from a random forest.

In either case, one might use a forward model selection process, i.e. one would first fit a model using only an intercept and then add predictors as long as that makes the bigger model significantly better (e.g. in terms of p-values) or substantially better (e.g. in terms of *AIC* or *BIC* values). For this, one might consider adding predictors in the order determined by the regression modeling process or by the variable importance scores of the random forest (as in Deshors & Gries, accepted pending revision, who apply this logic to results from a random forest fit with interaction predictors). Once the chosen model selection process is completed, the final model is visualized with, for instance, effects plots (Fox 2003) to see what it is that each predictor contributed to the random forest predictions.

If GSMs are applied to the present artificial data set (here as a backward selection example using `MASS::stepAIC`), the final model finds the interaction between P2 and P3 and thus achieves perfect accuracy and represents the effects faithfully – however, the results are less than ideal in terms of significance tests and confidence intervals because the didactically motivated perfect split/complete separation in the simulated data renders all coefficients insignificant – with real data, however, where perfect predictability is much less likely, this is correspondingly much less likely to happen.

In sum, GSMs seem to be gaining ground as more and more relatively hard-to-interpret (deep) machine learning methods are employed and researchers are struggling with making sense of what such black boxes return. While I have not seen them being used in linguistics, GSMs are an interesting alternative worth exploring and certainly more reasonable than trying to interpret a random forest with a single tree fit on all the data.

4 Concluding remarks

As the size and complexity of data sets in different subfields of linguistics grow, it becomes more and more important to study them with methods that do their complexity justice. While it is great that more and more linguists are using multifactorial methods to find patterns in their data, this also increases the risk of applications that raise more problems than they might solve. Tree-based methods have become a welcome alternative for data sets that defy regression-based methods especially in noisy and unbalanced corpus data, and that, in and of itself, is potentially a good thing. However, in this paper, I showed

- that there can be patterns in data that make trees underperform considerably when it comes to accuracy, variable importance/parsimony, and effects interpretation;
- that random/conditional inference forests can sometimes help with (some of) these issues, but that the way in which some studies try to interpret random forests – with a single tree – is also not ideal.

These issues are especially problematic given the otherwise positive trend that corpus-linguistic data and methods are now informing linguistic theorizing more than they have for a long time. In order to address these problems, I discussed several analytical possibilities including explicitly created interaction variables in trees and random forests and interpreting random forests on the basis of (1) (ideally multiple) representative trees, (2) effects plots, and (3) GSMs. This paper can obviously only stimulate discussion rather than settle the matter(s) at hand – in fact, it seems every single aspect of random forests is currently being lively discussed in bioinformatics journals: sampling of data (with or without replacement), splitting criteria (Gini vs. *p*-values), variable importance measures (error rate vs. permutation-based versus *AUC* [the latter two conditional or unconditional]), variable selection, whether random forests can capture or detect interactions in the presence of correlated predictors, imbalanced response variables, etc., all of which affect the (quality of the) results .... However, I hope that the above observations and suggestions lead to a greater awareness of the potential pitfalls of trees and forests, but also the opportunities they offer.

Acknowledgments: I am immensely grateful to three reviewers for *CLLT* (for this paper) and two reviewers from *Corpora* (who reviewed Deshors & Gries, to appear) as well as Tobias J. Bernaisch for their input and in particular Benedikt Heller for his comprehensive feedback that helped make this paper better; the usual disclaimers apply.

References

Bernaisch, Tobias, Stefan Th. Gries, & Joybrato Mukherjee. 2014. The dative alternation in South Asian English(es): Modelling predictors and predicting prototypes. *English World-Wide* 35(1). 7–31.

Boulesteix, Anne-Laure, Silke Janitza, Alexander Hapfelmeier, Kristel Van Steen & Carolin Strobl. 2015. Letter to the editor: On the term 'interaction' and related phrases in the literature on random forests. *Briefings in Bioinformatics* 16(2). 338–345.

Breiman, Leo. 2001. Random forests. *Machine Learning* 45. 5–32.

Crawley, Michael J. 2013. *The R Book*. 2nd ed. Chichester: John Wiley & Sons.

Dasgupta, Abhijit. 2014. Reprtree: Representative trees from ensembles. A package for R; GithubRepo: reprtree.

Deshors, Sandra C. & Th. Gries Stefan. Accepted pending revision. Mandative subjunctive vs. *should* in world Englishes: A new take on an old alternation. *Corpora*.

Dilts, Philip. 2013. Modelling phonetic reduction in a corpus of spoken English using random forests and mixed-effects regression. Edmonton: University of Alberta Unpublished Ph.D. dissertation.

Ellis, Nick C., Ute RöMer & O'Donnell Matthew Brook. 2016. *Usage-based approaches to language acquisition and processing: Cognitive and corpus investigations of construction grammar. Language learning*, vol. 66. (Suppl. 1, Language Learning Monograph Series). New York: John Wiley.

Forina, Michele, Monica Casale, Paolo Oliveru & Silvia Lanteri. 2009. CAIMAN brothers: A family of powerful classification and class modeling techniques. *Chemometrics and Intelligent Laboratory Systems* 96(2). 239–245.

Fox, John. 2003. Effect displays in R for generalised linear models. *Journal of Statistical Software* 8(15). 1–27.

Gries, Stefan Th. 2013. *Statistics for linguistics with R*, 2nd rev. and ext. edn, 359. Berlin & Boston: De Gruyter Mouton.

Gries, Stefan Th. & Stefanie Wulff. 2012. Regression analysis in translation studies. In Michael P. Oakes & Ji Meng (eds.), *Quantitative methods in corpus-based translation studies: A practical guide to descriptive translation research*, 35–52. Amsterdam & Philadelphia: John Benjamins.

Hansen, Sandra & Roman Schneider. 2013. Decision tree-based evaluation of genitive classification: An empirical study on CMC and text corpora. In Iryna Gurevych, Chris Biemann & Torsten Zesch (eds.), *Language processing and knowledge in the web*, 83–88. Berlin & New York: Springer.

Baayen, Harald R., Laura A. Janda, Tore Nesset, Anna Endresen & Anastasia Makarova. 2013. Making choices in Russian: Pros and cons of statistical methods for rival forms. *Russian Linguistics* 37(3). 253–291.

Hothorn, Torsten, Peter Bühlmann, Sandrine Dudoit, Annette Molinaro & Van Der Laan Mark. 2006a. Survival ensembles. *Biostatistics* 7(3). 355–373.

Hothorn, Torsten, Kurt Hornik & Achim Zeileis. 2006b. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics* 15(3). 651–674.

Hothorn, Torsten & Achim Zeileis. 2015. partykit: A modular toolkit for recursive partytioning in R. *Journal of Machine Learning Research* 16. 3905–3909.

Hundt, Marianne. 2018. It is time that this (should) be studied across a broader range of Englishes: A global trip around mandative subjunctives. In Sandra C. Deshors (ed.), *Modeling world Englishes: Assessing the interplay of emancipation and globalization of ESL varieties*, 217–244. Amsterdam & Philadelphia: John Benjamins.

Ishwaran, Hemant & Udaya B. Kogalur. 2019. randomForestSRC. R package version 2.8.0. https://cran.r-project.org/web/packages/randomForestSRC/index.html.

James, Gareth, Daniela Witten, Trevor Hastie & Robert Tibshirani. 2013. *An introduction to statistical learning with applications in R*. Berlin & New York: Springer.

Janitza, Silke, Carolin Strobl & Anne-Laure Boulesteix. 2013. An AUC-based permutation variable importance measure for random forests. *BMC Bioinformatics* 14. 119.

Jones, Zachary M. & Fridolin Linder. 2017. edarf: Exploratory data analysis using random forests. R package version 1.1.1. https://CRAN.R-project.org/package=edarf.

Klavan, Jane, Maarja-Liisa Pilvik & Kristel Uiboaed. 2015. The use of multivariate statistical classification models for predicting constructional choice in spoken, non-standard varieties of Estonian. *SKY Journal of Linguistics* 28. 187–224.

Kuhn, Max & Kjell Johnson. 2013. *Applied predictive modeling*. New York et al.: Springer.

Liaw, Andy & Matthew Wiener. 2002. Classification and regression by randomForest. *R News* 2(3). 18–22.

Molnar, Christoph. 2018. Interpretable machine learning: A guide for making black box models explainable. E-book at <https://christophm.github.io/interpretable-ml-book/>, version dated 10 Dec 2018.

Rezaee, Abbas Ali & Seyyed Ehsan Golparvar. 2017. Conditional inference tree modelling of competing motivators of the positioning of concessive clauses: The case of a non-native corpus. *Journal of Quantitative Linguistics* 24(2–3). 89–106.

Ripley, Brian. 2018. tree: Classification and regression trees. R package version 1.0-39. https://CRAN.R-project.org/package=tree.

Strobl, Carolin, Anne-Laure Boulesteix, Thomas Kneib, Thomas Augustin & Achim Zeileis. 2008. Conditional variable importance for random forests. *BMC Bioinformatics* 9. 307.

Strobl, Carolin, Anne-Laure Boulesteix, Achim Zeileis & Torsten Hothorn. 2007. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics* 8. 25.
Strobl, Carolin, James Malley & Gerhard Tutz. 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.
Szmrecsanyi, Benedikt, Jason Grafmiller, Benedikt Heller & Röthlisberger Melanie. 2016. Around the world in three alternations: Modeling syntactic variation in varieties of English. *English World-Wide* 37(2). 109–137.
Tagliamonte, Sali A. & R. Harald Baayen. 2012. Models, forests, and trees of York English: Was/were variation as a case study for statistical practice. *Language Variation and Change* 24(2). 135–178.
Therneau, Terry & Beth Atkinson. 2018. rpart: Recursive partitioning and regression trees. R package version 4.1-13. https://CRAN.R-project.org/package=rpart.
Tomaschek, Fabian, Peter Hendrix & R. Harald Baayen. 2018. Strategies for addressing collinearity in multivariate linguistic data. *Journal of Phonetics* 71. 249–267.
Wright, Marvin N. & Andreas Ziegler. 2017. ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software* 77(1). 1–17.
Wright, Marvin N., Andreas Ziegler & Inke R. König. 2016. Do little interactions get lost in dark random forests? *BMC Bioinformatics* 17. 145.
Zhu, Ruoqing, Donglin Zeng & Michael R. Kosorok. 2015. Reinforcement learning trees. *Journal of the American Statistical Association* 110(512). 1770–1784.

Bionote

Stefan Th. Gries

Stefan Th. Gries is Professor of Linguistics in the Department of Linguistics at the University of California, Santa Barbara (UCSB), Honorary Liebig-Professor of the Justus Liebig Universität Giessen (since September 2011), and since 1 April 2018 also Chair of English Linguistics (Corpus Linguistics with a focus on quantitative methods, 25%) at the Justus Liebig Universität Giessen.