Computational Statistics

Interactive Graphics for Visually Diagnosing Forest Classifiers in R

--Manuscript Draft--

Manuscript Number:	COST-D-21-00279R1
Full Title:	Interactive Graphics for Visually Diagnosing Forest Classifiers in R
Article Type:	S.I. : Latin American Conference on Statistical Computing 2021
Keywords:	Statistical Visualization, Interactive Visualization, Interpretable Machine Learning , Ensemble Model
Manuscript Classifications:	1.00560: Classification Analysis; 1.02050: Interactive Statistical Graphics; 1.02490: Machine Learning; 1.04520: Statistical Graphics
Corresponding Author:	Natalia da Silva, PhD Universidad de la Republica Facultad de Ciencias Economicas y de Administracion URUGUAY
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Universidad de la Republica Facultad de Ciencias Economicas y de Administracion
Corresponding Author's Secondary Institution:	
First Author:	Natalia da Silva, PhD
First Author Secondary Information:	
Order of Authors:	Natalia da Silva, PhD
	Dianne Cook, PhD
	Eun-Kyung Lee, PhD
Order of Authors Secondary Information:	
Funding Information:	
Abstract:	This article describes structuring data and constructing plots to explore forest classification models interactively. A forest classifier is an example of an ensemble since it is produced by bagging multiple trees. The process of bagging and combining results from multiple trees produces numerous diagnostics which, with interactive graphics, can provide a lot of insight into class structure in high dimensions. Various aspects of models are explored in this article, to assess model complexity, individual model contributions, variable importance and dimension reduction, and uncertainty in prediction associated with individual observations. The ideas are applied to the random forest algorithm and projection pursuit forest but could be more broadly applied to other bagged ensembles helping in the interpretability deficit of these methods. Interactive graphics are built in R using the ggplot2, plotly, and shiny packages.

Authors Response to Review

Our responses to the review are written in **blue**.

Response to Editor

We have received the reports from our advisors on your manuscript, "Interactive Graphics for Visually Diagnosing Forest Classifiers in R", Based on the advice received, the Editor feels that your manuscript could be accepted for publication should you be prepared to incorporate minor revisions. When preparing your revised manuscript, you are asked to carefully consider the reviewer comments which are attached, and submit a list of responses to the comments. Your list of responses should be uploaded as a file in addition to your revised manuscript.

Thank you. We appreciate the chance to do a further revision, and hope that these changes are satisfactory.

Response to Reviewer 2

We appreciate all of your feedback which help to improve our paper.

Summary of the paper

The paper introduces an app for visual exploration of tree-based ensemble models in three steps. First, the app enables the user to investigate how well the model performs when it comes to predicting each class in its observation-level functionality. Second, the tree-level section allows the user to select a tree based on a variable importance measure in one of the first three nodes and visualize the way the first three splits are made in that tree. Finally, the app visually compares the predictive performance of the model to a benchmark one, which is illustrated by comparing a projection-pursuit-based random forest to a standard one.

The paper and its accompanying app seem to be a good fit for Computational Statistics, as they apply interesting visualization techniques to a tree-based ensemble model that will surely prove useful for practitioners.

However, for this potential to materialize the authors should put more emphasis on the limitations of their approach. In particular, the app focuses on the first three nodes of the trees in question, which would be much less informative in settings requiring deep trees. **The app was modified to adress this issue. In the new application the user can select specific nodes to see (from one to all the nodes). If there are a lot of nodes will be better to select 4 or less each time to see better the complete display but the application is not limited to see just the first three nodes.**

Furthermore, plots such as the parallel coordinate plot would lose their appeal in settings with thousands of variables. A brief discussion of circumstances limiting the usability of the app would address these concerns and allow users to easily determine whether the app is suitable for their application.

The discussion has been extended to include limitations.

Substantive comments

1. p. 8, line 42: The issue of visualizing the tetrahedron is arguably not of first-order importance as it is not as helpful for classification problems with more than four classes. It might be more useful to focus

exclusively on the side-by-side jittered dotplot as this approach generalizes in a more straightforward way to problems with more classes.

As we mentioned in page 7 the ternary plot can be generalized to more than three dimensional space, with G classes, the ternary plot idea is generalized to a G-1 dimensional simplex. Then if we have 4 classes the generalized ternary plot is a tetrahedron as shown in Figure 5 but if we have more than four classes we can also generalize the ternary plot idea. In the fishcatch data used in this paper to illustrate the shiny app there are 7 classes and we generalized the ternary plot to a 6-dimensional simplex as is shown in the last panel of Figure 8 (6-didmensional simplex shown pairwise), using tourr package you can see better the 6-dimensional simplex in different rotations https://vimeo.com/707047875. The structure is complex because a 6-dimensional simplex has 7 vertices, 21 edges and 35 faces.

2. p. 11, line 16: As section 3 discusses all other plots produced by the app it would be useful if it also included an overview of the parallel coordinate plot that only appears in section 4, so the meaning of that plot is not immediately clear (e.g. what is displayed on the vertical axis).

   We have included more explanaition about the parallel coordinate plot in Section 4 where this figure appears and added a corresponding reference for further details. In the vertical axis the values for each variable are standardized to easily compare them.

3. p. 13, line 38: I find the choice of the interaction driver for the model tab puzzling. I would argue that the key statistic distinguishing the trees in the ensemble is OOB error so a more natural design choice would be to include jittered points in the OOB error boxplot so that the user can explore the well or poorly performing trees in more detail. I do not have a good intuition for why the user would be interested in exploring trees based on variable importance.

   The second tab was modified based on your helpful comments. We have included three main changes in the second tabset, first we have included jittered points in the OOB error boxplot where the user can select trees based in their performance. Second, we have included a panel to select the nodes to see in the importance variable plot, the mosaic plot and the density plot. The panel allows you to select more or less than three nodes and the mentioned plots are updated acordingly. Finally, we rearranged some of the plots to make the selection and interaction more intuitively presented in Figure 10.

4. p. 15, line 35: The OOB error plot is somewhat hard to read due to some classes being harder to classify than others, and a potential solution to that would be using a different scale on the vertical axis (e.g. percent of highest OOB error for a class).

   We explore different alternatives as you suggested but the result was not clearly better. We have removed the points and just keep thiner lines which improve the visualization. Also in the app, because the interactivity, it is easy to see the oob error values mousing over the plot which in the static version it is not possible.

Minor comments

1. p. 6, line 17: introducing the G_s notation is unnecessary as it is not used in the remainder of the paper (also, the meaning of the subscript s is not explained).

   I have removed G_s, as you notices it is unnecessary here

2. p.6, line 21: it should be "Alternatively, it can be computed" instead of "Alternatively, can be computed".

   Done

3. p. 6, line 33: it should be "The plot in the right" instead of "The plot at right".

   Done

4. p. 8, line 20: it should be "into a three-dimensional space" instead of "into three dimensional".

   Done

5. p.8, line 49: crabs are "classified", not "predicted".

   Done

6. p. 10, line 40: the phrase "one of the biggest difficulties for the app in order manage linking" should be rewritten to, for example, "one of the biggest difficulties when it comes to the app managing linking".

   Done

7. p. 12, line 46: using the phrase "some trees could be removed from the forest" instead of "some trees could be pruned from the forest" would make this sentence clearer.

   Done

8. p. 14, line 49: "a receiver" instead of "an receiver".

   Done

9. p. 16, line 32: it would be useful to include a legend of the class colors.

   Class color legends were not included in the interactive web app because the interactivity allows you to get the class name when you mouse over each plot.

10. p. 17, line 6: " will perform" instead of "will performs".

    Done

Response to Reviewer 4

I think the paper overall is good. The comments below are a combination of comments on the paper itself and some suggestions for the plotting in the software (which don't have to be changed for the paper). I think also that some of the details of the plots need to be explained better.

MAIN COMMENTS:

Section 4 in general: since the big point here is that the software is interactive, I think the discussion in this section would be a lot more useful if you illustrated step by step with screenshots an example where your interactive drill-down allows you to quickly discover things. For instance, say there are some outliers in plot A. Then you click on them and see some further details in plot B that help you understand what is going on. In this section you show a bunch of screenshots and label their capabilities, but you don't actually demonstrate an end-to-end example of the interactivity. For instance, in Figure 9 you give an example of a set of actions, why don't you actually show us the results and the changes that occur in a sequence of plots? Furthermore, I would show some actual examples of screenshots of what shows when you hover over observations.

In Section 4 we have added more details and examples of use and interpreting results to address your comment.

CONTENT COMMENTS:

1. Page 2: In describing Random Forests, in addition to randomly sampling variables to try to split on in each node, don't they also initially sample a subset of variables for each tree? If so, you should note what settings you use in this paper (how many variables are selected for each tree). Also, if the variables are randomly sampled at each node, rather than simply choosing the best projection possible for, say, any pair of variables, what effect does this have on the importance measures? What effect is there to determining the number of variables to use in each projection (e.g. 3 instead of 2)? Is this something you have experimented with? I think the paper could also benefit from a simple preliminary

example of, say, two features and two classes and showing the plot of the linear combination resulting from projection pursuit. You could even include an image from the original paper.

A random forest has two sources of randomness: variable sampling and bootstrap samples. The variable sampling is used for node partition, where each split is produced from a random subset of variables. There is not an initial sample of variables for each tree.

We did not formally explore the sensitivity of the random variable sample size for the measure of variable importance. In the algorithm you can select the proportion of variables to be sampled at each node. In the crab and fishcatch examples presented in the paper two variables were sampled at each node.

We have included an image from the original paper with the boundary that would result from a classification tree fitted using the PPtree algorithm to take into account your last request. The plot shows a simple example from simulated data with two features and three classes. The partitions generated by the PPtree algorithm are oblique to the axis, incorporating the association between the two variables.

2. Page 4 line 48: should O_k also have a subscript for the variable j? Also $\hat{y}_{ikj}$

   Subscript j for O_k is not needed because for every variable j O_k will be the same and not change depending on j, the same for $\hat{y}_{ik}$. In the only case subscript j can be included is in $\tilde{y}_{ik}$ to indicate the permuted variable but for simplicity we just keep the formula as it is in the original PPforest paper (https://www.tandfonline.com/doi/abs/10.1080/10618600.2020.1870480)

3. Page 5 Figure 1: You mention that the variables should be standardized for the coefficients to be interpretable as importances and be comparable. But in the upper right corner, the plots of the variables distributions, for instance for node 2, are not in the standardized range (node 2 has an x-axis that goes up to 5 and 6). **As you mentioned the data should be standardized for the coefficients to be interpretable as importances. We have fixed the density plot and the x axis accordingly.**

   Is the distribution being plotted here the value of the linear combination for each class, rather than a single variable? **Yes, the density plots shows the data projection at each node. This is now made clear in the caption.**

   Is there something going on here with the default x limits for doing KDE? **The x limits were fixed.**

   As a side note, you show that the first cut perfectly separates the blue and orange species. You may want to consider having the bounds for the KDE be the observed range of the linear combination for each class so that the density plots won't overlap in the plot for node 1, which they seem to do more than they should if the separation is perfect.

   We explore your suggestion (changing the bound for the KDE to observe range) but the result was not better so we have kept the original.

   Furthermore, if you have any control over the color scheme (if you don't I would recommend adding this as a variable), it might be a good idea in this paper to have the colors for the blue and orange species be blue and orange, and have differing shading patterns to distinguish male from female, or something similar; perhaps you can choose more similar colors within both species. Or, better yet, can you use different plotting symbols, say triangles vs circles to distinguish the species, and shaded vs hollow to distinguish the sex?

   The selected palette (Dark2) is a qualitative palette suited to mapping a categorical data because it is colorblind friendly. It is a little unfortunate that the categories in the crabs data are colors, which makes the explanation confusing. We have changed the category names to be simply O and B so that there is less confusion.

   Also, can you include labeling in your plot that the sub-plots are node ID (e.g., put the label "node" above the 3 density plots). **We have included in each sub-plots "Node" before the number and also update this in the app.**

What do the numbers in the grey boxes at the terminal nodes represent? **The numbers in the gray boxes are the predicted class, we fixed the plot to show the class names**

4. Figures 1 and 2: I see that the potential range for coefficient values for the projection pursuit seem to be between -1 and 1. I imagine this is a feature of the algorithm. Perhaps include in your description of projection pursuit a note about this fact, for those unfamiliar, and mention what optimization criteria is used in it.

   Yes, the coefficient values are between -1 and 1 because the length of the projection vector is 1. We find the linear combinations, a which maximize $\frac{a^T Ba}{a^T Wa}$, where B and W are the between and within group sums of squares. We get $\hat{a}_1 = e_1$ where $e_1 ... e_s$ are eigenvectors of $W^{-1}B$. In this case we find the 1-D projection optimizing the LDA index. We have included in Section 3 LDA and PDA indexes which are optimized to get the linear projections to make it more clear how the coefficients are computed.

5. Page 6: If you use a notation for VI for global variable importance (you define it as the average across k trees), I'd just define it here as an equation and use it in the text (e.g in line 19).

   Formulas for all VI were defined in equations and mentioned in the text. It is clearer now.

6. Page 6 line 48: does MDS need a citation? Also, can you connect the results in this discussion to the RF graphics, e.g. in terms of same variables with high RF importance being important in MDS? Otherwise, this whole discussion seems unnecessary. If you are doing it simply to understand the data, I would move this section to right after you introduce the data, before you show the other graphics. Here, it seems like a tangent. Also, are these MDS and tetrahedral plots part of your software or do you perform them separately? I would say so explicitly in either case.

   A reference for MDS has been included. In Section 3 we describe the mapping of diagnostics to visualizations. MDS is presented in Subsection 3.3 (Similarity of cases) the key connection with the forest is that we are using the proximity matrix information which is one of the diagnostics described for PPforest or RF and we use this information to compute the MDS. This visualization can be useful to identify in which cases the model is performing well and which it is having problems.

   MDS and generalized ternary plot (tetrahedral plot in case there are four clases) are both implemented in the app and we have added a comment in the paper to clarify this.

7. Page 7 Figure 2: why does the left plot need to be a dot plot rather than, say, a single box or violin plot for each variable? I don't understand what role the x axis for each variable plays here. If it doesn't play any role (e.g. is just for the jittering), then don't use it.

   We selected a jittered side-by-side dot plot in this case instead of a box plot because we want to see the values for each tree, and it makes it more sensible when linking observations between multiple plots. Jittering spreads the data on the categorical variable axis to better read the distribution.

8. Page 9, Figure 4: is the dimension of the simplex always at most 3 or G-1? What would it look like with more classes? I'd clarify whether this only works for your example or whether it can always be drawn.

   he dimension of the simplex is $G - 1 = 3$ (Figure 5 revised version), since in crab data there are 4 classes. Ternary plot can be generalized to more than three dimensional – with $G > 4$ classes, the ternary plot idea is generalized to a $G - 1$ dimensional simplex. In the fishcatch data there are 7 classes and the ternary plot to a 6-dimensional simplex as it is shown in pairwise plots in Figure 6. Using the tourr package you can see the 6-dimensional simplex in a rotation through 6D, see https://vimeo.com/707047875. The structure is complex because a 6-dimensional simplex has 7 vertices, 21 eddges and 35 faces. This has been clarified in the paper.

9. Page 10, Figure 5: similar remark as before. I think a violin plot or something similar for each class, rather than a jittered dot plot, is more correct. The x-axis for each class is a "fake" axis. I think jittered dot plots should only be used with scatterplots to make sure individual dots can be distinguished.

   We are using points because it is important to see the individual cases and the jitter is to overcome overplotting on the categorical variable. It is not recommended using jittered scatterplots because it would change the real values of your data.

10. Page 10, line 40: This sentence is muddled "As Sievert (2020) describes, one of the biggest difficulties for the app in order manage linking between plots is the data structure management for each widget". Furthermore, what does "widget" refer to here? Use a more descriptive term.

    At their core Shiny widgets are mini-applications created using the shinyApp function. We have changed the language of the paragraph to help with this terminology.

11. Page 10, line 45: If you are just going to demonstrate the interactivity, wouldn't it be better to just continue with the original crab dataset, or have one dataset throughout? Unless there is something particularly interesting about this dataset.

    Yes, fishcatch data are more difficult to classify and then more interesting to show some of the results. For simplicity to describe the main plot of the app we use crab data but to describe the app we prefer to present a more complex problem, more classes, and umbalanced data.

12. Page 12 Figure 6: in the enhanced PCP, you say all variable pairs are shown at least once as neighbors. Is there a way, either through ordering or adding dividing lines, etc. to help a reader find the pair they are interested in? Or is there some sorting done so that, the pairs are ordered from left to right in terms of some metric of "interesting-ness"? There seems to be no order in the given plot. Do you automatically pick the number of dimensions to show in MDS? Should the tetrahedral plots here have some kind of axis labels? See Figure 7 where they do.

    A PCP can be ordered in different ways, and we have modified the PCP to include variable order based on the mean value. As you suggested, we have also change the grid lines to keep only the vertical grid in gray. In the MDS we only used two dimensions but can be more if deemed more appropriate for a particular problem. We have included labels in the generalized ternary plot.

13. Page 13 line 46: I wasn't aware that there was an option in RF to prune under-performing trees (as opposed to nodes within a tree) in an ensemble. Please elaborate or include a reference. Also, in this case, maybe a different word than "prune", such as "discard".

    This comment is in page 12, In the original RF there is not an option to prune under-performing trees. The comment is just to mention a possible use for the information we get from the app, also we are presenting the app using PPforest and this can be a possible extension for our algorithm.

14. Page 16 Figure 10: in the upper LH corner, where you select the classes, do these appear as text that you type or delete, or as buttons you can toggle on and off? Seems like the latter would be better. Also, the bottom two plots have different x axes, so it is hard to compare them (if they should be compared, the axes should be the same; perhaps you could even plot them on the same plot?). Furthermore, there are no titles for either, and they have the same axis labels so I have no idea what the difference is between them.

    In the upper LH corner we are using the function 'selectizeInput()' from shiny R package which creates a select list that can be used to choose a single or multiple items from a list of values, you do not have to type the class names you just select the names from a list. For the bottom two plots (variable importance plots) we have fixed the axis and include titles in each plot, so it is easier to compare them.

STYLISTIC COMMENTS

1. Should out of bag be written OOB rather than oob?

   We have changed to using OOB now.

2. Page 3 line 29: when you say "proportionately predicted", do you mean something specific, like that the class probabilities are proportionate to their frequency in the data?

   Cases that are predicted to be multiple classes, without a clear majority in any of them, indicate that those cases are difficult to classify. The paragraph was not clear and we modify it to:" Cases that are predicted to be multiple classes, with similar class probabilities, indicate difficult to classify observations".

3. Page 3 line 38: These two sentences don't seem to work together: "Each model uses samples of variables. So that, the accuracy of the models can be compared when the variable is included or omitted." Do you mean "Thus" rather than "So that"?

   Yes, should be Thus. It is fixed.

4. Page 4: "Specie" in Figure 1 caption is misspelled.

   Done

5. Page 11 line 12: For the sentence beginning "The data feeding. . . ", separate each clause with semicolons rather than commas.

   Done

6. Page 11 line 39: say "ID" or "index" rather than "id" ; same for, say, page 13 line 44

   Done

7. Page 11 line 49: I would modify the final sentence to say that "all variable pairs are shown next to each other once" rather than "all variables are neighbors".

   Done

Noname manuscript No.
(will be inserted by the editor)

Interactive Graphics for Visually Diagnosing Forest Classifiers in R

Natalia da Silva · Dianne Cook · Eun-Kyung Lee

Received: date / Accepted: date

Abstract This article describes structuring data and constructing plots to explore forest classification models interactively. A forest classifier is an example of an ensemble since it is produced by bagging multiple trees. The process of bagging and combining results from multiple trees produces numerous diagnostics which, with interactive graphics, can provide a lot of insight into class structure in high dimensions. Various aspects of models are explored in this article, to assess model complexity, individual model contributions, variable importance and dimension reduction, and uncertainty in prediction associated with individual observations. The ideas are applied to the random forest algorithm and projection pursuit forest but could be more broadly applied to other bagged ensembles helping in the interpretability deficit of these methods. Interactive graphics are built in R using the `ggplot2`, `plotly`, and `shiny` packages.

Keywords Statistical Visualization · Interactive Visualization · Interpretable Machine Learning · Ensemble Model

Natalia da Silva
Instituto de Estadística (IESTA), Universidad de la República
E-mail: natalia.dasilva@fcea.edu.uy

Dianne Cook
Department of Econometrics and Business Statistics, Monash University
E-mail: dicook@monash.edu

Eun-Kyung Lee
Department of Statistics, Ewha Womans University
E-mail: lee.eunk@gmail.com

1 Introduction

The random forest (RF) algorithm (Breiman 1996) was one of the first ensemble classifiers developed. It combines the predictions from individual classification and regression trees (CART) (Breiman et al. 1984) built by bagging observations (Breiman 1996) and random predictor sample at each tree node. These produce diagnostics in the form of uncertainty in predictions for each observation, importance of variables for the prediction, predictive error for future samples based on out-of-bag (OOB) case predictions, and similarity of observations based on how often they group together in the trees.

Ensemble classifiers have grown in popularity (Dietterich 2000, Talbot et al. 2009), and the basic ideas behind the random forest can be applied to virtually any type of model. The benefits for classification are reduced variability in predictive error, and the suite of diagnostics provides the potential for better understanding the class structure in the high-dimensional data space. The use of visualization on these diagnostics, in association with multivariate data plots, completes the process to better understand the underlying problem.

A conceptual framework for model visualization can be summarized in three strategies: (1) visualize the model in the data space, (2) look at all members of a collection of a model, and (3) explore the complete process of model fitting (Wickham et al. 2015a). The first strategy is to explore how well the model captures the data characteristics (the model in the data space), which contrasts with determining if the model assumptions hold (data in the model space). The second strategy is to look at a group of models instead of only the best model. This strategy can offer a broad understanding of the problem by comparing and contrasting possible models. The last strategy focuses on exploring the process of the model fitting in addition to the end result.

There has been some, but not a abundant, research on visualizing classification models. Urbanek (2008) presents interactive tree visualization implemented in the java software called `KLIMT` that includes zooming, selection, multiple views, interactive pruning, and tree construction as well as the interactive analysis of forests of trees using treemaps. Cutler and Breiman (2011) developed a java package called `RAFT` to visualize a forest classifier, that included variable selection, parallel coordinate plots, heat maps, and scatter plots of some diagnostics. Linking between plots is limited. Quach (2012) presents interactive forest visualization using the R package `iPlots eXtreme` (Urbanek 2011) where several displays are shown in one window with some linking between them available. Silva and Ribeiro (2016) describes visualizing components of an ensemble classifier.

This article describes structuring interactive graphics to facilitate visual exploration of ensemble classifiers using RFs and projection pursuit random forests (PPF) (da Silva et al. 2021) as examples. The PPF algorithm builds on the projection pursuit tree (PPtree) (Lee et al. 2013) algorithm which uses

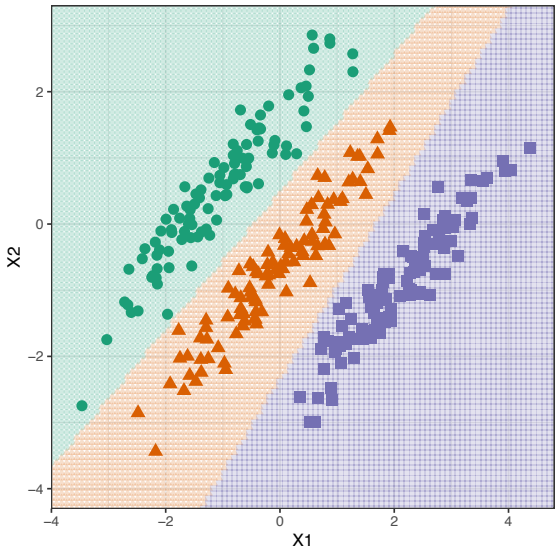

Fig. 1: Decision boundaries for PPtree algorithm on 2D simulated data. The partitions generated by PPtree algorithm are oblique to the axis, incorporating the association between the two variables.

projection pursuit at each tree node to find the best linear combination of variables to separate the classes. Figure 1 shows the boundary that would result from a classification tree fitted using the PPtree algorithm (Example from (da Silva et al. 2021)).

The visualization approach is consistent with the framework in Wickham et al. (2015a), and the implementation is built on the newest interactive graphics available in R (R Core Team 2016). The purpose is to provide readily available tools for users to explore and improve ensemble fits and obtain an intuition for the underlying class structure in data. Interactive plots are a key component for model visualization that helps the user see multivariate relationships and be more efficient in model diagnosis. Multiple levels of data are constructed for exploration: observation, model, and ensemble summaries. The visualization tools presented in this article help open up black-box models and contribute to the interpretability of these methods.

Two data set are used in this paper, crab (Campbell and Mahon 1974) and fishcatch data (Puranen 2017). To illustrate the mapping of diagnostics to visualization crab data are used because of its simplicity. The second data set is introduced to illustrate the interactive web app which combines the visualizations introduced in this paper and present a more complex problem to explore within the application.

The article is structured as follows. Section 2 describes the ensemble components to be assessed. Section 3 maps the ensemble components to the visual elements. The web app is described in Section 4 and further work is discussed in Section 5.

2 Diagnostics in Forest Classifiers

The diagnostics typically available are:

- ***Out-of-bag error:*** For each model in the ensemble, some cases of the original data are not used. Predicting the response for these cases gives a better estimate for the error of the model with future data. The OOB error rate is a measure for each model that is combined in the ensemble and is used to provide the overall error of the ensemble.

- ***Uncertainty measure for each observation:*** Across individual (classification) models we can compute the proportion of times that a case is predicted to be each class. If a case is always predicted to be the true class, there is no uncertainty about an observation. Cases that are predicted to be multiple classes, with similar class probabilities, indicate difficult to classify observations. They may be important by indicating neighborhoods of the data space that would benefit from a more complex model, or simply, measurement errors in the data.

- ***Variable importance:*** Each model uses samples of variables. Thus, the accuracy of the models can be compared when the variable is included or omitted. There are several versions of statistics that use this to provide a measure of the variable importance for prediction.

- ***Similarity measure for pairs of observations:*** In each model, a pair of observations will be either in the same terminal node or not. This is used to compute a proximity matrix. Cluster analysis on this matrix can be used to follow up the classification to assess the original labeling. It may suggest improvements or mistakes in original labels.

In addition to these overall ensemble statistics, each individual model has its own diagnostics, measuring error, variables utilized, and class predictions. Visualization will enable the individual models to be examined, relate these to the data and their contribution to the ensemble.

3 Mapping Ensemble Diagnostics to Visual Components

This section describes the mapping of diagnostics to visualizations. These are illustrated using the Australian crabs data (Campbell and Mahon 1974). The data has 200 cases, 5 predictors, and 4 classes (combinations of species and sex, BM, BF, OM, and OF). The predictors are: FL (the size of the frontal lobe length, in mm), RW (rear width, in mm), CL (length of mid-line of the carapace, in mm), CW (maximum width of carapace, in mm), BD (depth of the body; for females, measured after displacement of the abdomen, in mm). This is old data, but it provides a good illustration of the visual methods.

3.1 Individual Models: PPtree

The PPF is composed of individual projection pursuit trees. PPtree algorithm uses a multi-step approach to fit a multi-class model by finding linear combinations to split on based on projection pursuit algorithm. Two projection pursuit indexes are used to find projections that separate classes, LDA (Lee et al. 2005) and PDA (Lee and Cook 2010).

The LDA index is defined as follows:

$$\mathbb{I}_{LDA}(\mathbf{A}) = \begin{cases} 1 - \frac{|\mathbf{A}^T\mathbf{W}\mathbf{A}|}{|\mathbf{A}^T(\mathbf{W}+\mathbf{B})\mathbf{A}|} & \text{for } |\mathbf{A}^T(\mathbf{W}+\mathbf{B})\mathbf{A}| \neq 0 \\ 0 & \text{for } |\mathbf{A}^T(\mathbf{W}+\mathbf{B})\mathbf{A}| = 0 \end{cases} \quad (1)$$

where $p \times q$ matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_q]$ defines an orthonormal projection from p-dimensions onto a q-dimensional subspace, $\mathbf{B} = \sum_{g=1}^{G} n_g(\bar{\mathbf{x}}_g - \bar{\mathbf{x}})(\bar{\mathbf{x}}_g - \bar{\mathbf{x}})^T$ is the $p \times p$ between-group sum of squares, $\mathbf{W} = \sum_{g=1}^{G} \sum_{i \in H_g} (\mathbf{x}_i - \bar{\mathbf{x}}_g)(\mathbf{x}_i - \bar{\mathbf{x}}_g)^T$ is the $p \times p$ within-group sum of squares where $H_g = \{i | y_i = g, i = 1, ..., n\}$, $\bar{\mathbf{x}}_g$ is the group mean vector and $\bar{\mathbf{x}}$ the overall mean vector. If the LDA index value is high, there is a large difference between classes.

The PDA index is useful when n is small relative to p and when the variables are highly correlated. In these situations the maximum likelihood variance-covariance matrix estimator will be close to singular, affecting the inverse calculation. The PDA index adjusts the variance-covariance matrix calculation, as follows:

$$\mathbb{I}_{PDA}(\mathbf{A}, \lambda) = 1 - \frac{|\mathbf{A}^T\mathbf{W}_{PDA}(\lambda)\mathbf{A}|}{|\mathbf{A}^T(\mathbf{W}_{PDA}(\lambda) + \mathbf{B})\mathbf{A}|} \quad (2)$$

where the notation is analogous to the LDA index, with the addition of $\lambda \in [0, 1)$ which is a shrinkage parameter, and a different within group sum of squares, $\mathbf{W}_{PDA}(\lambda) = \text{diag}(\mathbf{W}) + (1 - \lambda)\text{offdiag}(\mathbf{W})$.

Figure 2 shows a visual ensemble of plots of a tree model on the crab data using LDA index (Equation 1) and two variables in each node split. There are three nodes for the four class problem. The nodes of this tree are based on projections of the data and their coefficients form the building block to calculate the variable importance. The density plot displays the data projection at each node, and the mosaic plot shows the confusion matrix for the nodes. The package `PPtreeViz` provides visual tools to diagnose a PPtree model. The PPF builds on these and modifies a little. The PPtree model is simpler than a regular classification tree because the classes are mostly separated by combinations of variables – just three projections are needed to see the differences between the four classes.

3.2 Variable Importance

The PPF algorithm calculates variable importance in three ways: permuted importance using accuracy, and two importance measures based on projection coefficients on standardized variables.

The first importance measure is comparable with the measure defined in the classical random forest algorithm. Importance variable is computed based on the OOB cases for the tree k (O_k) for each j predictor variable. Permuted importance of the variable j in the tree k is defined in Equation 3.

$$VI_{jk}^{(1)} = \frac{\sum_{i \in O_k} I(y_i = \hat{y}_{ik}) - I(y_i = \tilde{y}_{ik})}{\#O_k} \tag{3}$$

where $\hat{y}_{ik}$ is the predicted class for the observation i in the tree k, and $\tilde{y}_{ik}$ is the predicted class for the observation i in the tree k after permuting the values for variable j. The global permuted importance measure is the average importance over all the trees in the forest. This measure is based on comparing the accuracy of classifying OOB observations using the true class with permuted (nonsense) classes.

The coefficients of each projection are examined to define the other two importance measures. If the variables have been standardized, the magnitude of these values indicates importance. For a single PPtree the variable importance is computed by a weighted sum of the absolute values of the coefficients across nodes, then the weights take the number of classes in each node into account (Lee et al. 2013).

The global variable importance in a PPforest then can be defined in different ways. One way is to average the variable importance from each PPtree across all the trees in the forest (Equation 4).

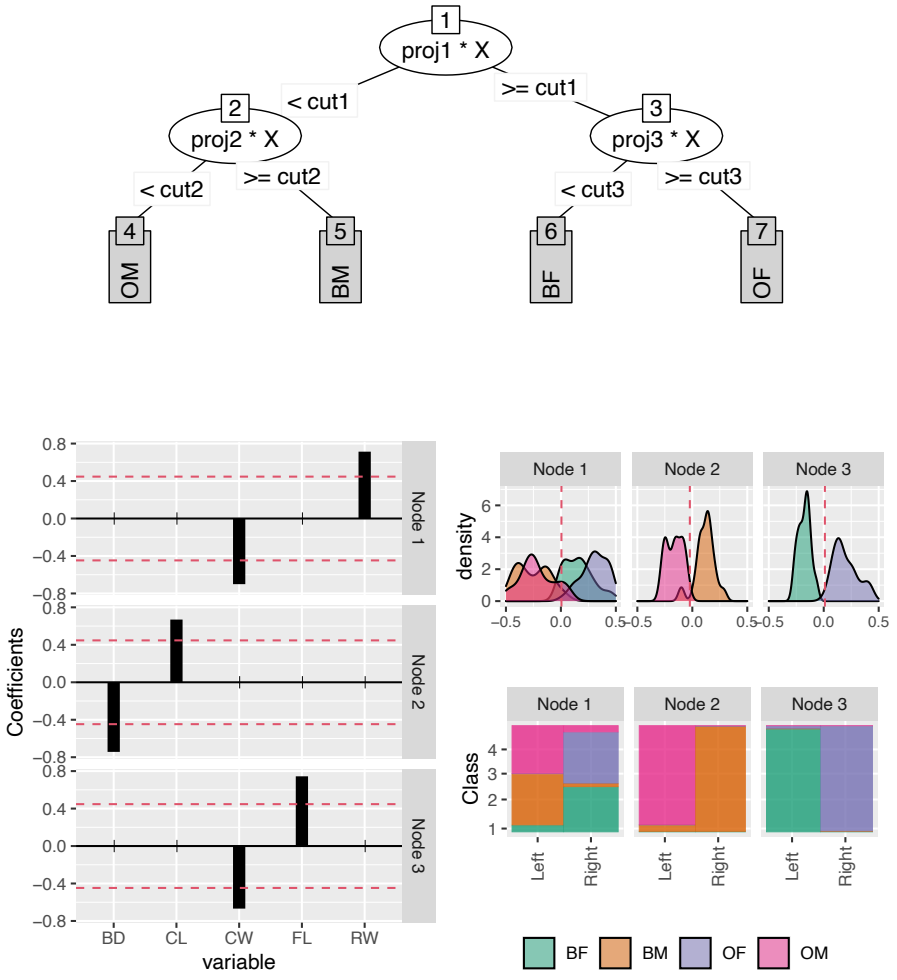

Fig. 2: Visualizing the PPtree model of the crab data. The tree has three nodes (top). The density plots show the data projections at each node, colored by group (middle). The dashed vertical red line indicates the split value of each node. Node 1 separates the sexes. Node 2 separate males from species and node 3 separate females from species. Mosaic plots of the confusion table for each split (bottom). Node 1 shows the clear split of sexes, with a small number of misclassifications. Node 2 where BM are separated from OM with small number of misclassifications. Node 3 where BF are separated from OF, almost perfect classification.

$$VI_j^{(2)} = \frac{1}{B}\sum_{k=1}^{B}\sum_{s=1}^{S}\frac{|a_{sjk}^{**}|}{G_s} \tag{4}$$

where a_{sjk}^{**} is the projected coefficient for node s and variable j in the k^{th} PPtree, and S is the total number of node partitions in the k^{th} tree. Note that $\sum_{s=1}^{S}\frac{|a_{sjk}^{**}|}{G_s}$ is the importance of the variable j in the PPtree k.

Alternatively, it can be computed as a weighted mean of the absolute value of the projection coefficients across all nodes in every tree as shown in Equation 5.

$$VI_j^{(3)} = \frac{1}{B}\frac{1}{G-1}\sum_{k=1}^{B}\sum_{s=1}^{S}(1-e_k)I_{sjk}|a_{sjk}^{**}| \tag{5}$$

where e_k is the OOB error rate of tree k, I_{sjk} is the projection pursuit index value of node s (da Silva et al. 2021).

Figure 3 shows the absolute projection coefficient of the top three nodes for all the trees in a forest model. This information is displayed by a side-by-side jittered dot plot. The red dots correspond to the absolute coefficient values for the tree model of Figure 2. The forest was built using random samples of two variables for each node, hence there are two coefficients for each node. At node 1 uses CW and RW with similar contribution. The scatterplot at right shows these two variables and the resulting boundary between groups that this would produce. Node 2 uses BD and CL where BD contributes the most to the separation. The plot in the right shows the boundary that is induced. Node 3 uses CW and FL with an even contribution by the two variables. For each tree in the forest, different decision rules are defined; the resulting boundaries on the previous plots are based on Rule 1 = $\frac{m_1}{2} + \frac{m_2}{2}$, where m_1 and m_2 are the mean of the left and right groups at each node.

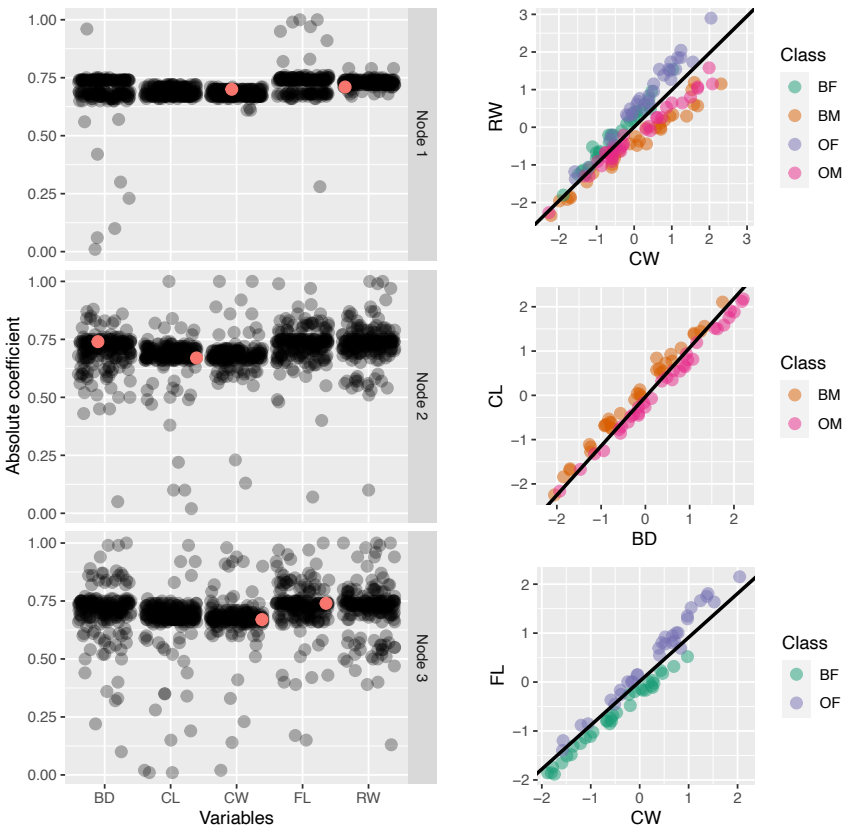

Fig. 3: Visualizing the variable importance of all the trees in the forest, for three nodes. Every particular node of each tree has an importance value for the corresponding variable. The values for the whole forest are displayed using a side-by-side jittered dot plot. The importance values are the absolute values of the projection coefficients. The red points correspond to these values for the tree shown in Figure 2. Two variables are randomly selected at each node for creating the best projection, and split. The plots at right show the variables used and the split made at each of the nodes of this tree.

3.3 Similarity of Cases

For each tree, every pair of observations can be in the same terminal node or not. Tallying this up across all trees in a forest gives the proximity matrix, an $n \times n$ matrix of the proportion of trees that the pair share a terminal node. It can be considered as a similarity matrix.

Multidimensional scaling (MDS) is used to reduce the dimension of this matrix, to view the similarity between observations (Kruskal 1964). MDS transforms the dataset into a low-dimensional space where the distances are approximately the same as in the full n dimensions. With G groups, the low-dimensional space should be no more than $G-1$ dimensions. Figure 4 shows the MDS plots for the three dimensional space induced by the four groups of the crab data. Color indicates the true species and sex. For this data, two dimensions are enough to see the four groups separated quite well. Some crabs are clearly more similar to a different group, especially in examining the sex differences. MDS is included in the interactive web app presented in Section 4.

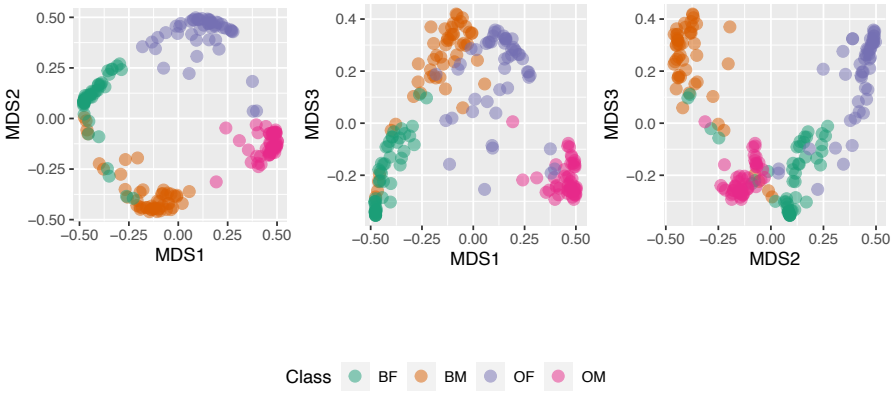

Fig. 4: Examining similarity between cases, using pairwise plots of multidimensional scaling into a three-dimensional space. It can be seen that most cases are grouped closely with their class, and particularly that the two species are distinct. There is more confusion of cases between the sexes.

3.4 Uncertainty of Cases

The vote matrix ($n \times G$) contains the proportion of times each observation was classified to each class while OOB. Two approaches to visualize the vote matrix information are used.

A ternary plot is a triangular diagram used to display compositional data with three components. More generally, compositional data can have any number of components, say p, and hence is constrained to a $(p-1)$ dimensional

simplex in p-space. The vote matrix is an example of compositional data with G components.

With G classes, the ternary plot idea is generalized to a $(G-1)-dimensional$ simplex (Sutherland et al. 2000; Schloerke et al. 2017). This is one of the approaches used to visualize the vote matrix and it is included in the interactive web app presented in Section 4.

For the crab data, $G = 4$ and the generalized ternary plot will be a tetrahedron. Figure 5 shows the tetrahedron structure for the crab vote matrix shown in three pairwise views. With well-separated classes, the colored points will each concentrate into one of the vertices. This is close but not perfect, indicating some crabs are commonly incorrectly classified. The `tourr` package (Wickham et al. 2011), tour methods for multivariate data visualization, can be used to view the animated version available for this example in `https://vimeo.com/170522736`.

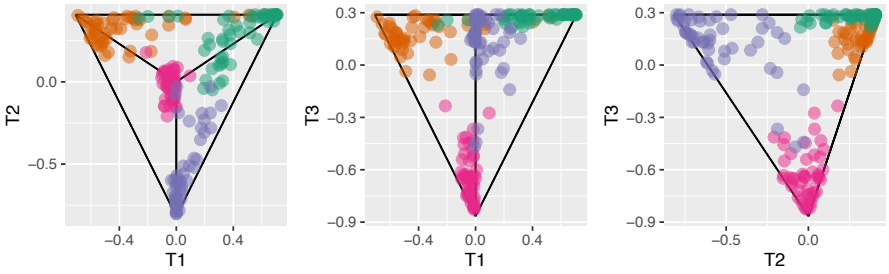

Fig. 5: Generalized ternary plot ((G-1) dimensional simplex, in this case a tetrahedron) representation of the vote matrix for four classes. The tetrahedron is shown pairwise. Each point corresponds to one observation and color is the true class. This is close but not a perfect classification since the colors are concentrated in the corners and there are some mixed colors in each corner.

Because visualizing the vote matrix with a $(G-1)$ dimensional tetrahedron requires dynamic graphics, a low-dimensional option is also provided. For each class, every individual case has a value between 0-1. A side-by-side jittered dotplot is used for the display, where the class is displayed on one axis and proportion is displayed on the other. For each dotplot, the ideal arrangement is that points are concentrated at 0 or 1 and only at 1 for their true class. This data is close to the ideal but not perfect, e.g. there are a few BM crabs (orange) that are frequently predicted to be BF (green), and a few BF crabs predicted to be another class.

Fig. 6: Another representation of the vote matrix, as a jittered side-by-side dotplot. It is not as elegant as the ternary plot, but it is useful because it places focus on each group. Each dotplot shows the proportion of times the case was predicted into the group, with 1 indicating that the case was always predicted to the group and 0 being never. On each dotplot, a single color dominates the top, indicating fairly clear distinctions between classes. Crabs from the B species, both male and female, have more uncertainty in predictions, as seen by more crabs from other classes having higher vote proportions than is seen in the O species.

4 Interactive Web App

Interaction is added to the plots described in Section 3 as well as other plots, which are organized into an interactive web app using `shiny` (Chang et al. 2015; Wickham 2021) for exploring the ensemble model. The app is orga-

nized into three tabs: individual cases; models; and performance comparison, to provide a model diagnostic tool. Interaction is provided as mouse-over labeling, mouse-click selection and brushing, with results linked across multiple plots. The app takes advantage of new tools provided in the `plotly` package (Sievert et al. 2017; Sievert 2020), developed as a part of Sievert's PhD thesis research (Sievert 2017).

The `plotly` functions directly translate a static `ggplot2` object by extracting information and storing it in JavaScript Object Notation (JSON). This information is passed as input to a javascript function to produce a web graphic. Interactions in a single display and links between different graphics are two key tasks an interactive visualization should accomplish (Xie et al. 2014).

As Sievert (2020) describes, one of the biggest difficulties when it comes to the app is managing linking between plots (also called a widget in the plotly documentation) requiring careful data structure management. Each plot has it own data structure and interaction. Putting them into the structure of a shiny app facilitates access to the widget data and coordinates selections across multiple plots.

To illustrate the shiny app characteristics, a different dataset (fishcatch) are used which presents a more challenging classification problem and then more interesting results to explore the app. The fishcatch dataset (Puranen 2017) contains 159 observations, with 6 physical measurement variables, and 7 types of fish, all caught from the same lake (Laengelmavesi) near Tampere in Finland. There are 35 bream, 11 parkki, 56 perch, 17 pike, 20 roach, 14 smelt and 6 whitewish. The shiny app showing fishcatch data can be accessed at `https://natydasilva.shinyapps.io/shinyppforest`.

4.1 Individual Cases

This tab is designed to examine uncertainty in the classification of observations and to explore the similarity between pairs of observations. The data feeding the display is an $n \times p$ data frame; containing the original data; the model statistics generated from the full $n \times G$ vote matrix along with its generalized ternary coordinates; and the first two MDS projections of the proximity matrix.

The plots in the tab are (1) a parallel coordinate plot (PCP) of the data, (2) the MDS display of the proximity matrix, (3) side-by-side jittered dotplot, and (4) generalized ternary plot of the vote matrix. Each of these plots are interactive in the sense that each one presents individual interactions (mouse-over) and they are linked so that selections in one display are propagated to other plots (clicking and selecting).

The diagram in Figure 7 illustrates the data pipeline (Buja et al. 1988; Wickham et al. 2009) for the interactive graphics in the case level tab. Solid

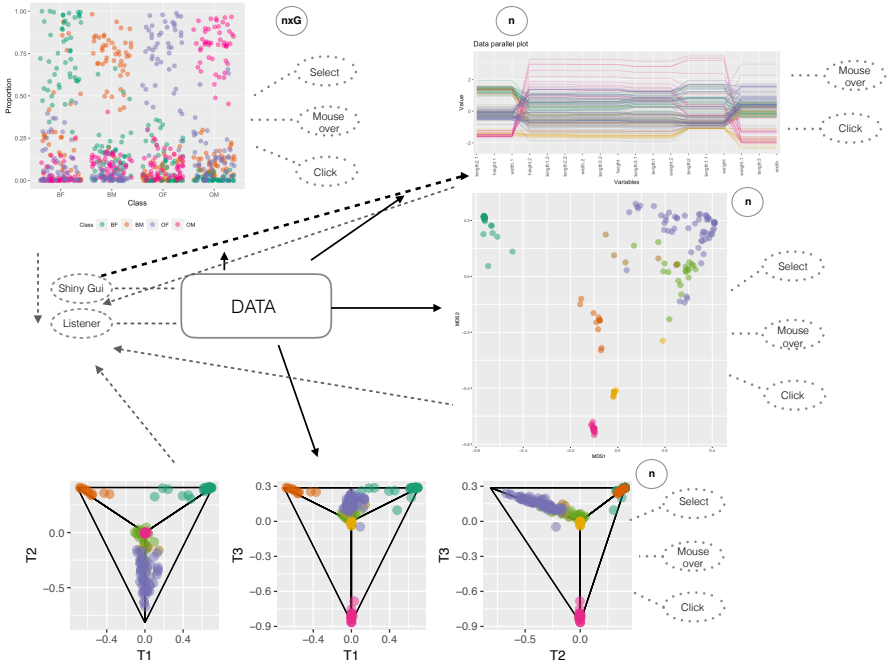

Fig. 7: Schematic diagram illustrating the interactivity in and between plots for individual level exploration panel of the web app. From the data, the four plots are generated. Each plot has a listener attached which collects user actions. When a user selects a point or line in any of the displays, it makes a change in the data which propagates updates to each of the plots.

lines indicate notifications from the source data to the plots, and dashed lines indicate notification of user action on the plot, which notifies the data source of actions to take. The data table is a reactive object that has a listener associated with it. Each of the plots is reactive and has numerous listeners. When users make selections on a plot, either by clicking or group selection, a change to the data is made in terms of an update on the selected cases. This invokes a note to other plots to re-draw themselves. The linking between plots is effectively one-to-one, based on the row ID of the data. The side-by-side jittered dotplot has $n \times G$ points, but selection can only be done within a dotplot. Selecting in one of the dotplots notifies the data table of the selection which triggers a re-draw of the other dotplots. Mouseovers on the plot pull additional information about the point or line under the cursor but do not link between plots.

Figure 8 shows the arrangement of plots in this first tab (the case level tab).

This selection of plots enables aspects of the model, relating to performance for individual cases, to be examined in the data space. The data plot is an essential element following the *model-in-the-data-space* philosophy of Wickham et al. (2015a). The choice was made to use a parallel coordinate plot because it provides a space-efficient display of the data. Alternatives include the tour, a dynamic plot, or a scatterplot matrix. Theoretically, either of these could be substituted or added. Parallel coordinate plots are useful to visualize multivariate numerical data. As is shown in the first plot in Figure 8, in a PCP each variable is placed equidistant and perpendicular to the x-axis, on the y-axis the values for each variable are standardized to easily compare them. Each observation is represented as a line connected across all the variables. In a PCP many variables can be compared together and seeing the relationships between them, identify and characterize groups of observations. The features can be ordered so that there are not too many lines intersecting resulting in an unreadable chart (Inselberg 2009). Two alternatives can be selected in shiny to draw the parallel coordinate plot: parallel or enhanced. Parallel draws the classic PCP and enhanced draws a modified version where variables are repeated (Hurley and Oldford 2011). Because reading a PCP is really only possible for neighboring variables, the variables are repeated so that all variable pairs are shown next to each other once.

Figure 9 shows the arrangement of plots in this first tab for selected points to illustrate how the tab works. In the MDS plot, the yellow points (smelt) which are grouped were selected, in the parallel plot it is clear all these observations are similar with small values in all the variables and small variability. These are easy points to classify for the model, in the side-by-side jittered dot plot all these points have a probability of being classified as the correct class (smelt) close to one, and in the ternary plot, all these points are concentrated in the vertices.

4.2 Models

This second tab in the app focuses on teasing apart the forest to examine the qualities of each tree. For each tree, information on the variable importance, the projections used at each node, and the OOB error is available. The data feeding into this tab is a list of models along with the original data frame. The tree ID is displayed when we mouse over the jittered side-by-side plot. This information is useful because, based on the accuracy, some trees could be removed from the forest outside of the app.

Figure 10 is a screenshot of the models tab. There are five plots, with varying levels of interaction: (1) a boxplot of OOB error for all trees including the data as jittered points, (2) a jittered side-by-side dotplot showing variable

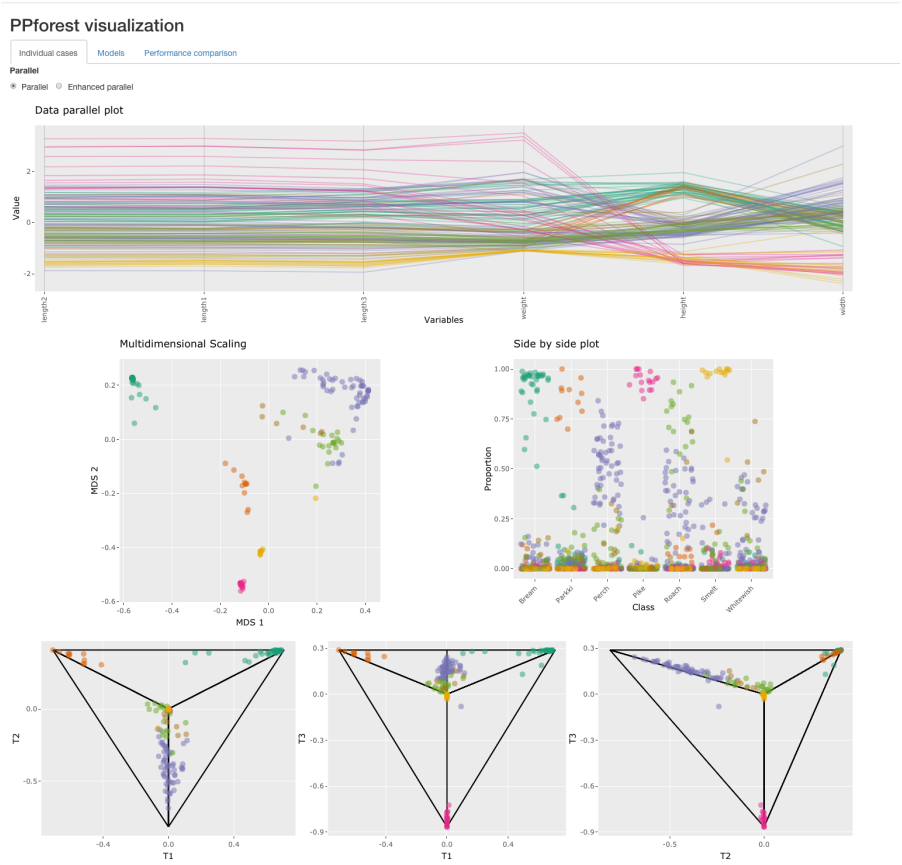

Fig. 8: Entry page for the web app, focusing on examining cases, in terms of similarity and uncertainty. The top plot shows the data, the remaining plots show similarity of cases and uncertainty in predictions. All of the plots are linked so that selecting elements of one will generate changes in the others.

importance for the top three nodes of all trees in the forest, (3) a static display of one PPtree, (4) a faceted density plot of projected data at each node of the tree, with split point indicated by a vertical line, and (5) a mosaic plot showing the confusion matrix for each node of the tree. The interaction is driven in three ways, from the boxplot – when the user selects a point in that display, the corresponding importance variable plot, tree plot from the `PPtreeViz`, tree density displays and mosaic plots are drawn. The tree plot from the `PPtreeViz` is used to visualize the selected tree structure. In addition, the variable importance values are highlighted for each variable for the top three nodes, by default, and the OOB error value for the selected tree on the boxplot (both in red). Additionally it is possible to select specific nodes and update the importance variable plot, density plot and mosaic plot (nodes can

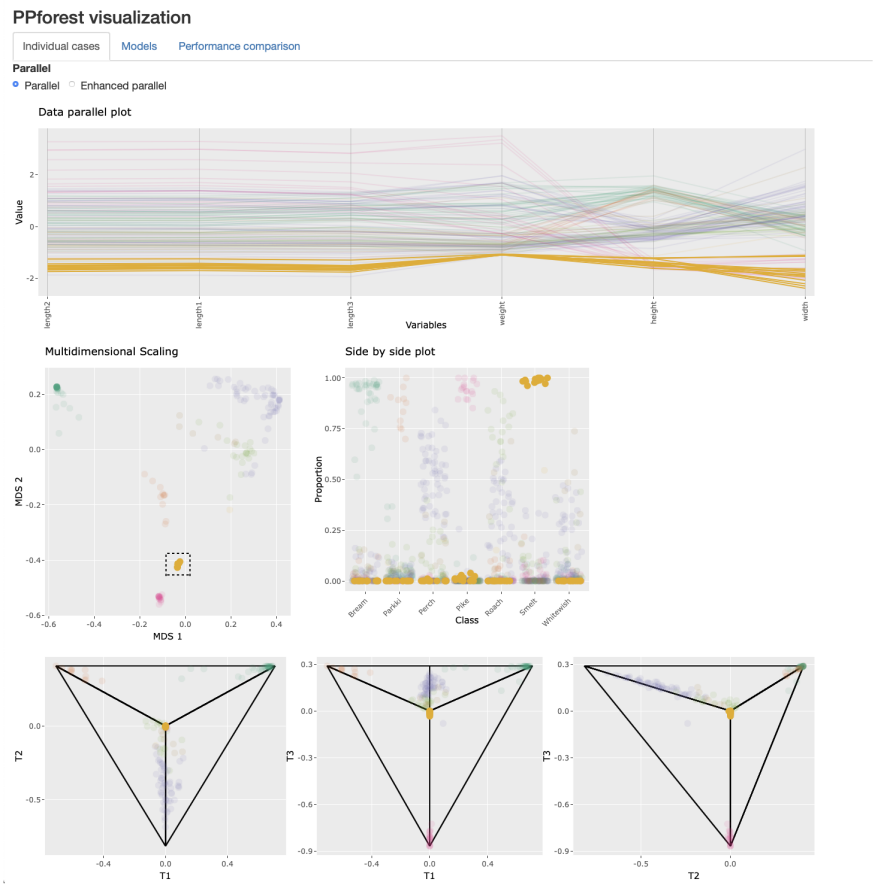

Fig. 9: Entry page for the web app, focusing on examining cases, for selected points.

be selected on the right corner on this second tab) and finally the interaction can be driven from the variable importance plot.

The selected tree in Figure 10 has very small OOB error shown as red point in the boxplot. Two variables were use in each node partition, the first four nodes were selected and shown in the importance variable plot, density plot and mosaic plot. In the importance variable plot `height` and `lenght1` were used in the first node partition being `height` the important variable to separate Bream and Parkki (Green and orange in density and mosaic plots) from the rest. In the second node `length1` and `length2` were both evenly important to separate Bream from Parkki. A similar analysis can be done for other nodes. This tab allow us to understand individual trees in the forest and analyzed which variables are important for the node partition related with the OOB error involved in that specific tree.

Fig. 10: The individual model tab in the web app. The OOB error is displayed in a boxplot including jittered points for all trees in the forest. Variable importance is displayed as jittered dot plots for selected nodes of all trees. The boxplot is linked to a display of the PPTree, the variable importance plot, and a density plot of the data projections showing splits at each selected nodes and confusion tables as mosaic plots. Clicking a point in the OOB error boxplot triggers various updates: each of the importance values for the same tree are highlighted (red), the tree that this corresponds to is drawn, the error for the tree is shown on the boxplot (in red), and the density plot and the mosaic plot are also updated. Also specific nodes can be selected to update the plots.

The diagram in Figure 11 illustrates the data pipeline for interactive graphics. The data source is a `PPforest` object. Interaction is driven in different ways, by the boxplot including jittered points. Selecting a point triggers a change in the data which cascades to re-draws of the other displays. Each plot has some information available on mouse over. Also selecting points in the variable importance triggers a change in the data and re-draws the displays corresponding to a specific tree. Finally selecting nodes update the data and re-draws the importance variable plot, density plot and mosaic plot.

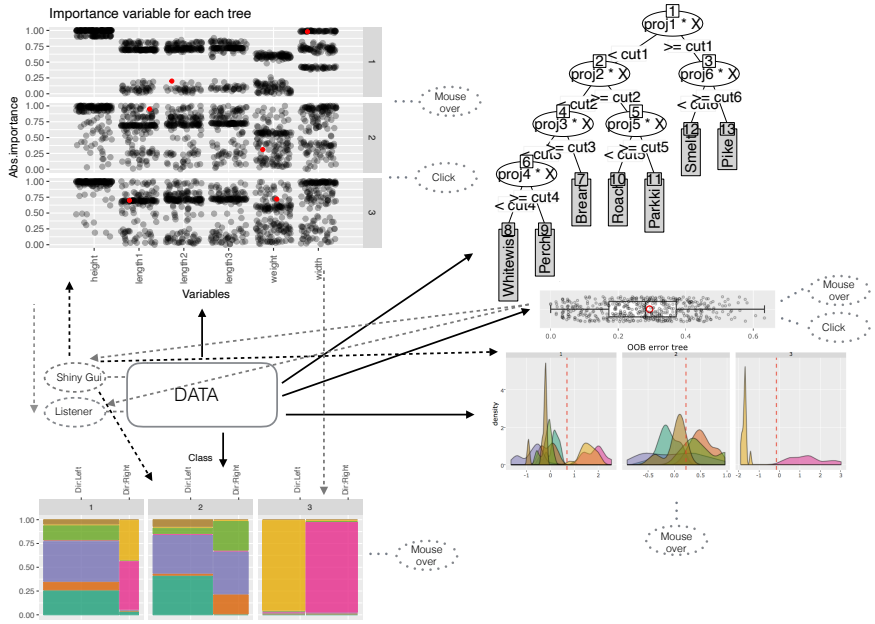

Fig. 11: Schematic diagram illustrating the interactivity in and between plots for model level exploration panel of the web app. The boxplot and the dotplot of variable importance has click selection, which invokes changes of the corresponding displays. Selecting a point in the boxplot update the node tree information (Shiny Gui) and update all the plots for the three first nodes, also nodes can be selected and update the corresponding plots. Finally selecting a point in the importance variable plot makes a change in the data, which propagates the importance values for other variables in this tree to be highlighted (red), draws the tree, highlights the error value of the tree, and shows the projections and confusion matrix for the three top nodes.

4.3 Performance Comparison

The third tab (Figure 12) examines the PPF fit and compares the result with a RF fit. There are four displays for each type of model: (1) Variable importance for all trees in the forest (same as in the models tab), (2) a receiver operating characteristic curve (ROC) comparing sensitivity and specificity for each class, (3) OOB error by the number of trees to assess complexity, (4) overall variable importance. There is very little interaction on this tab. Users can select to focus on a subset of classes or choose the importance measure to show. Being able to focus on class can help to better understand how well the model performs across classes and the focus can be especially useful for unbalanced data. Examining the OOB error by trees enables an assessment of how few trees might be used to provide an equally accurate prediction of future data.

The ROC is used to summarize the trade-off between sensitivity and specificity. The plot shows the sensitivity and specificity when a parameter of the classifier is varied (Hastie et al. 2011). The specificity and sensitivity were computed with the `pROC` package. If more than two classes are available a multi-class ROC analysis is needed. Several solutions have been proposed for multi-class ROC. Some of the proposed reduced the multi-class problem to a set of binary problems. The approach used for a multi-class ROC analysis in this article is called one-against-all (Allwein et al. 2000).

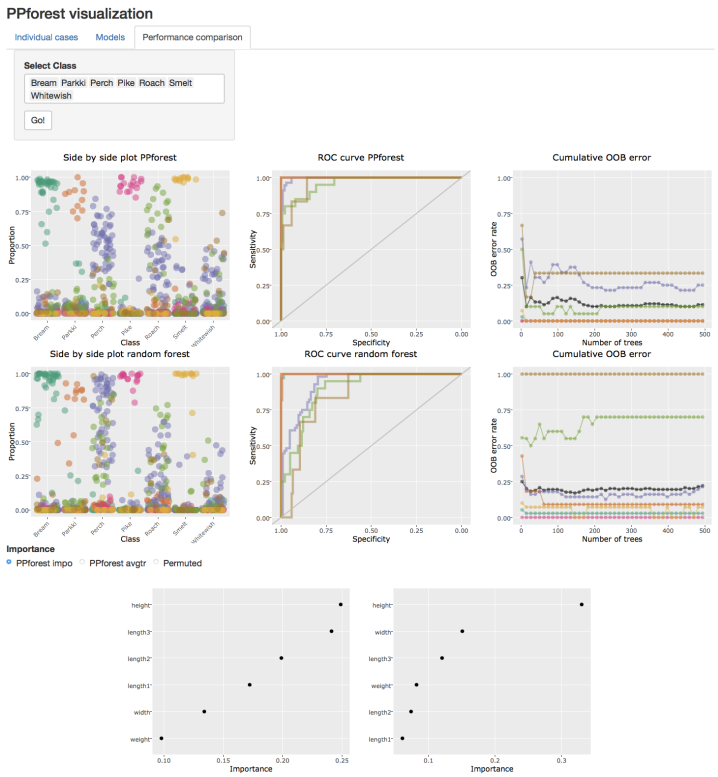

Fig. 12: Performance comparison tab of the web app. ROC curves displaying sensitivity against specificity for the classes are shown, along with the OOB error by the number of trees used to build the forest, and overall variable importance. Displays are shown for the PPF and RF, for comparison purposes. Users can select a class to focus on, using the text entry box.

5 Discussion

Having better tools to open up black box models will provide a better understanding of the data, the model strengths and weaknesses, and how the model will perform on future data. This visualisation app provides a selection of interactive plots to diagnose PPF models. This shell could be used to make an app for other ensemble classifiers. The philosophy underlying the collection of displays is "show the model in the data space" explained in Wickham et al. (2015a). It is not easy to do this, and to completely take this on would require plotting the model in the p-dimensional data space. In the simplest approach, as taken here, it means to link the model diagnostics to displays of the data. Then it is possible to probe and query to obtain a better understanding such

as finding regions in the data that prove difficult to fit, and detract from the predictive accuracy, or that don't adhere to model assumptions.

The app is implemented using the technology for interactive graphics provided by the `plotly` package. It was one of the first to experiment with the application of these tools. One challenge with plotly is that when layers containing different data are created in a ggplot2, it is difficult to specify the unique keys required for linking with other plots.

There are some limitations with providing functionality through the app, and generally with the displays provided. Successful apps make it easier for the user to do common tasks, and this is the design principal here. Only limited functionality is provided to the user, but the code is available and a knowledgeable data analyst might use this as a basis for extending the methods as appropriate for their data. Performance of the app is good for relatively small data sets. If the number of observation is large, overploting will be an issue in some of the visualizations. If the number of variables is large, the parallel coordinate plot and plot of variable importance would benefit by the addition of a user input to select variables.

There are many possible extensions to the app that could help it to be a tool for model refinement:

1. Using the diagnostics to weed out under-performing models in the ensemble.
2. Identifying and boosting models that perform well, particularly if they do well for problematic subsets of the data.
3. Problematic cases could be removed, and ensembles re-fit.
4. Classes as a whole could be aggregated or re-organised as suggested by the model diagnostics, to produce a more effective hierarchical approach to the multiple class problem.

Working within the R environment makes all of these possible when using command line outside the app, given that the unique IDs of models and cases can be exported from the app.

The app has helped to identify ways to improve the PPtree algorithm and consequently, the PPF model which especially apply to multiclass problems. Multiple splits for the same class would enable nonlinear classifications. Split criteria tend to place boundaries too close to some groups, due to heteroskedasticity being induced by aggregating classes. Forests are not always better than their constituent trees, and if the trees can be built better, the forest will provide better predictions.

6 Supplementary Materials

This article was written with the R packages knitr (Xie 2015), ggplot2 (Wickham July 2009), and dplyr (Wickham et al. 2015b), and the files to reproduce the article and results is available at https://github.com/natydasilva/PPforestViz.
The code to reproduce the shiny app is available at https://github.com/natydasilva/shinyPPforest.

References

Allwein, E. L., Schapire, R. E., and Singer, Y. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of machine learning research*, 1(Dec):113–141, 2000.

Breiman, L. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. Classification and regression trees. wadsworth. *Belmont, CA*, 1984.

Buja, A., Asimov, D., Hurley, C., and McDonald, J. A. Elements of a viewing pipeline for data analysis. pages 277–308. 1988.

Campbell, N. A. and Mahon, R. J. A multivariate study of variation in two species of rock crab of genus *leptograpsus*. *Australian Journal of Zoology*, 22:417–425, 1974.

Chang, W., Cheng, J., Allaire, J., Xie, Y., and McPherson, J. shiny: Web application framework for r, r package version 0.11. 2015.
URL `http://CRAN.R-project.org/package=shiny`

Cutler, A. and Breiman, L. Raft: Random forest tool. 2011.
URL `http://www.stat.berkeley.edu/~breiman/RandomForests/cc_graphics.htm`

da Silva, N., Cook, D., and Lee, E.-K. A projection pursuit forest algorithm for supervised classification. *Journal of Computational and Graphical Statistics*, pages 1–21, 2021.

Dietterich, T. G. *Ensemble methods in machine learning*, pages 1–15. Springer Verlag, New York, 2000.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2011.

Hurley, C. B. and Oldford, R. Eulerian tour algorithms for data visualization and the pairviz package. *Computational Statistics*, 26(4):613–633, 2011.

Inselberg, A. *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications. 233 Spring Street, New York, NY 10013*. USA: Springer Dordrecht Heidelberg London New York, 2009.

Kruskal, J. B. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

Lee, E.-K. and Cook, D. A projection pursuit index for large p small n data. *Statistics and Computing*, 20(3):381–392, 2010.

Lee, E.-K., Cook, D., Klinke, S., and Lumley, T. Projection pursuit for exploratory supervised classification. *Journal of Computational and Graphical Statistics*, 14(4), 2005.

Lee, Y. D., Cook, D., Park, J.-w., Lee, E.-K., et al. Pptree: Projection pursuit classification tree. *Electronic Journal of Statistics*, 7:1369–1386, 2013.

Puranen, J. Finland fish catch. `https://ww2.amstat.org/publications/jse/jse_data_archive.htm` 2017.

Quach, A. T. Interactive random forests plots. 2012.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016.
URL `https://www.R-project.org/`

Schloerke, B., Wickham, H., Cook, D., and Hofmann, H. Escape from boxland: Generating a library of high-dimensional geometric shapes. *The R Journal*, `https://journal.r-project.org/archive/accepted` 2017.

Sievert, C. Interfacing r with the web for accessible, portable, and contents interactive data science. 2017.

Sievert, C. *Interactive web-based data visualization with R, plotly, and shiny*. CRC Press, 2020.

Sievert, C., Parmer, C., Hocking, T., Chamberlain, S., Ram, K., Corvellec, M., and Despouy, P. *plotly: Create interactive web-based graphs via plotly's API*, 2017. R package version 1.1.0.
URL https://github.com/ropensci/plotly

Silva, C. and Ribeiro, B. *Visualization of individual ensemble classifier contributions*, pages 633–642. Springer International Publishing, Cham, 2016.

Sutherland, P., Rossini, A., Lumley, T., Lewin-Koh, N., Dickerson, J., Cox, Z., and Cook, D. Orca: A visualization toolkit for high-dimensional data. *Journal of Computational and Graphical Statistics*, 9(3):509–529, 2000.

Talbot, J., Lee, B., Kapoor, A., and Tan, D. S. Ensemblematrix: Interactive visualization to support machine learning with multiple classifiers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*, pages 1283–1292. Association for Computing Machinery, New York, NY, USA, 2009.

Urbanek, S. Visualizing trees and forests. In C. Chen, W. Härdle, and A. Unwin, editors, *Handbook of Data Visualization*, Springer Handbooks of Computational Statistics, chapter III.2, pages 243–264. Springer, 2008.

Urbanek, S. iplots extreme: next-generation interactive graphics design and implementation of modern interactive graphics. *Computational Statistics*, 26(3):381–393, 2011.

Wickham, H. *Mastering shiny*. " O'Reilly Media, Inc.", 2021.

Wickham, H. *ggplot2: Elegant graphics for data analysis*. useR. Springer, July 2009.

Wickham, H., Cook, D., and Hofmann, H. Visualizing statistical models: Removing the blindfold. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 8(4):203–225, 2015a.

Wickham, H., Cook, D., Hofmann, H., Buja, A., et al. tourr: An r package for exploring multivariate data with projections. *Journal of Statistical Software*, 40(2):1–18, 2011.

Wickham, H., Francois, R., and Rstudio. `dplyr`: A grammar of data manipulation. http://cran.r-project.org/web/packages/dplyr/index.html, 2015b. Maintained by Wickham, H.

Wickham, H., Lawrence, M., Cook, D., Buja, A., Hofmann, H., and Swayne, D. F. The plumbing of interactive graphics. *Computational Statistics*, 24(2):207–215, 2009.

Xie, Y. *Dynamic documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition, 2015. ISBN 978-1498716963.
URL http://yihui.name/knitr/

Xie, Y., Hofmann, H., Cheng, X., et al. Reactive programming for interactive graphics. *Statistical Science*, 29(2):201–213, 2014.

Electronic Supplementary Material