treeheatr: an R package for interpretable decision tree visualizations

This manuscript (permalink) was automatically generated from trang1618/treeheatr-manuscript@5908eb4 on May 2, 2020.

Authors

- **Trang T. Le**
  0000-0003-3737-6565 · trang1618 · trang1618
  Department of Biostatistics, Epidemiology and Informatics, Institute for Biomedical Informatics, University of Pennsylvania, Philadelphia, PA 19104

- **Jason H. Moore**
  0000-0002-5015-1099 · EpistasisLab · moorejh
  Department of Biostatistics, Epidemiology and Informatics, Institute for Biomedical Informatics, University of Pennsylvania, Philadelphia, PA 19104 · Funded by National Institutes of Health Grant Nos. LM010098 and AI116794.

Abstract

Summary

treeheatr is an R package for creating interpretable decision tree visualizations with the data represented as heatmaps at the tree's terminal nodes. Going beyond the if-then step-by-step logic of a decision tree, the inclusion of a heatmap offers a broader view of the classification or regression problem and provides meaningful clarification of different node splits in the tree. Working harmoniously with other packages, *treeheatr* empowers the user with refined controls over the statistical threshold and presentation of the tree and heatmap.

Availability and implementation

The *treeheatr* package is freely available under the permissive MIT license at https://trang1618.github.io/treeheatr. It comes with a detailed vignette that is automatically built with GitHub Actions continuous integration.

Contact

ttle@pennmedicine.upenn.edu

Supplementary information

Supplementary data are available at Bioinformatics online.

Introduction

Tree-based algorithms such as random forests and gradient boosted trees are widely used techniques that comprise an important section of supervised machine learning. Visualizing and intepreting their building blocks, the single decision trees, are the first steps toward understanding these complex tree-based structures. However, it is difficult to incorporate the tree's predictive performance and the feature space in a single visualization. Existing softwares frequently treat all nodes in a decision tree similarly, leaving limited options for improving information presentation at the leaf nodes. Specifically, state-of-the-art libraries such as Python's dtreeviz, while producing aesthetic trees with detailed histograms at inner nodes, draw pie chart at leaf nodes. The *ggparty* R package allows the user to have full control of the representation of each node but fixes the terminal node widths, which can limit the ability to show more collective visualizations.

We have developed the *treeheatr* package to utilize the leaf node space to show the data as a heatmap where the samples and features are optionally clustered to improve interpretation. After simple installation, the user can apply *treeheatr* on their classification or regression problem with a single function:

```
heat_tree(data, task = 'classification', target_lab = 'Outcome')
```

This one line of code above will generate the conditional inference tree, perform clustering, and produce a decision tree-heatmap as a *ggplot* object that can be viewed in RStudio's viewer pane, saved to a graphic file, or embedded in an RMarkdown document. This example assumes a classification problem, but one can also apply *treeheatr* on a regression problem by changing the `task` argument.

This article is organized as follows. In Section 2, we show an example *treeheatr* application by employing its functions on a real-world clinical dataset from a study of diabetes mellitus in a high risk population of Pima Indians [1]. In Section 3, we describe in details the important functions and corresponding arguments in *treeheatr*. We demonstrate the flexibility the user has in tweaking these arguments to enhance understanding of the tree-based models applied on their dataset. Finally, we discuss general guidelines for creating effective decision tree-heatmap visualization.

A simple example

This example visualizes the conditional inference tree model built to predict whether or not a patient has diabetes from a dataset provided by the National Institute of Diabetes and Digestive and Kidney Diseases [1]. This dataset of 768 female patients at least 21 years old of Pima Indian heritage near Phoenix, Arizona was downloaded from Kaggle and has eight features: age, number of pregnancies, plasma glucose concentration, diastolic blood pressure, skin fold thickness, 2-hour serum insulin, body mass index (BMI) and diabetes pedigree function. Detailed descriptions of these variables and data source can be found on the Kaggle page.

The following lines of code computes and visualizes the conditional decision tree along with the heatmap containing features that are important for building this model (Fig. 1):

```r
heat_tree(
  data = diabetes,
  target_lab = 'Diabetes status',
  label_map = c(`0` = 'Negative', `1` = 'Positive')
)
```

The `heat_tree()` function takes a data frame, a character string indicating the column name associated with the outcome/phenotype (e.g., Diabetes status) and other optional arguments such as the mapping of the outcome label.

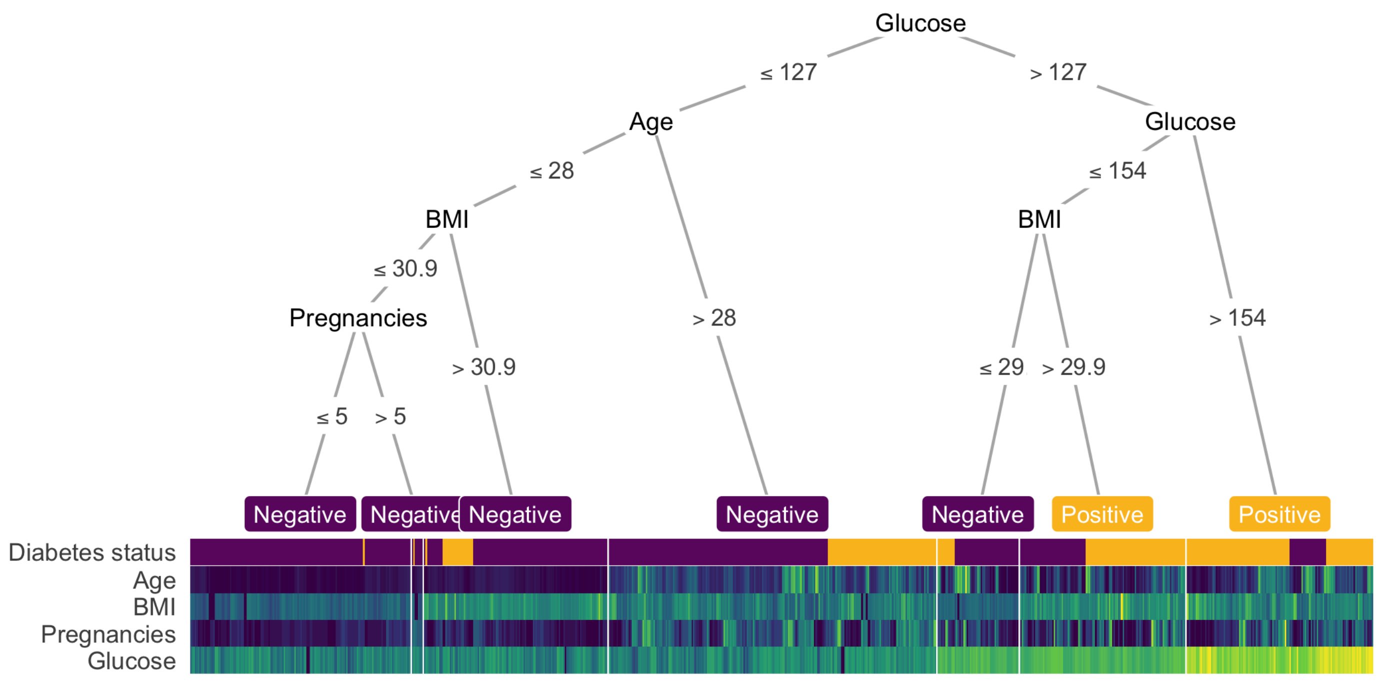

Figure 1: A decision tree-heatmap for predicting whether an individual has diabetes.

In this example, we observe that glucose level is the first determining factors in predicting diabetes status. When this value is above 127 but not larger than 154 (observations with light green glucose value), BMI helps further distinguish the group with diabetes from the other. On the left branches, while these samples are predicted to not have diabetes by majority voting, the leaf nodes have different purity. These seemingly non-beneficial splits present an opportunity to teach machine learning novices the different measures of node impurity such as the Gini index or cross-entropy [2]. General patterns in the heatmap for this example are admittedly difficult to parse because of the large number of observations, but we can still observe similar color patterns between age and the number of pregnancies indicating a correlation between these two features, which is expected.

Methods

Conditional decision trees [3] are nonparametric models performing recursive binary partitioning with well-defined theoretical background. Conditional trees support unbiased selection among covariates and avoid overfitting problems, producing competitive prediction accuracy [3]. *treeheatr* utilizes the *ggparty* R package to compute the conditional tree for a classification or regression problem (indirectly via the *partykit* R package) along with its edge and node information.

While *ggparty* assumes fixed terminal widths, *treeheatr* employs a flexible node layout to accommodate the different number of samples shown in the heatmap at each terminal node. This new node layout structure supports various terminal node widths, prevents crossings of different tree branches, and generalizes as the trees grow in size. This new layout weighs the *x*-coordinate of the parent node according to the level of the child nodes in order to avoid crossing of tree branches. This relative weight can be adjusted with the `lev_fac` parameter in `heat_tree()`. `lev_fac = 1` sets the parent node's *x*-coordinate perfectly in the middle of those of its child nodes. The default `lev_fac = 1.3` seems to provide aesthetically pleasing trees independent of the tree size (see vignette). The user can define a customized layout for a specific set of nodes and combine that layout with the automatic layout for the other nodes.

As default, *treeheatr* automatically performs clustering when organizing the heatmap. To order the features, clustering is run separately on the two groups of features, continuous and categorical, across all samples (including the outcome label, unless `clust_target = FALSE`). To order the samples, clustering is run on samples within each terminal node of all features. *treeheatr* uses the `daisy()` function in the *cluster* R package with the Gower metric [4] to compute dissimilarity in both continuous and nominal categorical feature types. We note that, while there is no definitive guideline for proper weighting of features of different types, the goal of the clustering step is to improve our interpretability of the tree-based model and not to make precise inference about each cluster. Therefore, *treeheatr* does not draw dendrograms and allows for the inclusion of outcome labels in clustering the samples.

In a visualization, it is difficult to strike the balance between enhancing understanding and overloading information. We believe showing a heatmap at the terminal node space provides additional information of the data in an elegant way that is not overwhelming and may even simplify the model's interpretation. We left it for the user to decide what type of information to be displayed at the inner nodes via different *geom* objects (e.g., `geom_node_plot`, `geom_edge_label`, etc.) in the *ggparty* package. For example, one may choose to show the distribution of the features and how they split the samples at these decision nodes, or each feature's corresponding Bonferroni-adjusted *P* values computed in the conditional tree algorithm [3].

Striving for simplicity, *treeheatr* utilizes direct labeling to avoid unnecessary legends. For example, in classification, the leaf node labels have colors corresponding with different classes, e.g., purple for Negative and yellow for Positive diabetes status (Fig. 1). As for feature values, the color scale legends may be misleading because these features may have been rescaled or normalized. As defaults, lighter colors are associated with higher values. This information can also be acquired from examining the edge labels. Specifically, in Fig. 1, high glucose values (larger than 154 on the rightmost branch) can be easily mapped to samples with light yellow color in the last row.

The integration of heatmap nicely complements the current techniques of visualizing decision trees. Node purity, a metric measuring the tree's performance, can be visualized from the distribution of true outcome label at each terminal node in the first row. Comparing these values with the terminal node label gives a visual estimate of how accurate the tree predictions are. Further, without specifically choose two features to show in a 2-D scatter plot, we can infer correlation structures among features in the heatmap. The additional clustering may also reveal sub-structures within a leaf node.

Conclusion

In this paper, we presented the new type of integrated visualization of decision trees and heatmaps, which provides a comprehensive data overview as well as model interpretation. We demonstrated that this integration uncovers meaningful patterns among the predictive features and highlights the important elements of conditional trees including feature splits and several leaf node characteristics such as prediction value, impurity and number of leaf samples. Implemented in an easily installed package with a detailed vignette, {treeheatr} can be a useful teaching tool to enhance students' understanding of this fundamental model before diving into more complex tree-based machine learning methods. This package has been released as open source software expected to receive contribution from other developers.

Future works on *treeheatr* include enhancements such as supporting heatmap visualization of a holdout set and highlighting the tree branches that point to a specific holdout sample. Simple data preprocess beyond scaling and normalizing the column may be beneficial for handling outliers and missing data points and thus constructing more robust models and more informative visualizations.

Acknowledgements

The *treeheatr* package was made possible by leveraging integral R packages including *ggplot2* [5], *partykit* [6], *ggparty* and many others. We would also like to thank Daniel Himmelstein for his helpful comments on the package's continuous integration configuration.

References

1. **Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Mellitus**
   Jack W. Smith, J. E. Everhart, W. C. Dickson, W. C. Knowler, R. S. Johannes
   Proceedings of the Annual Symposium on Computer Application in Medical Care (1988-11-09)
   https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2245318/
   PMCID: PMC2245318

2. **The elements of statistical learning: data mining, inference, and prediction**
   Trevor Hastie, Robert Tibshirani, J. H. Friedman
   Springer (2009)
   ISBN: 9780387848570

3. **Unbiased Recursive Partitioning: A Conditional Inference Framework**
   Torsten Hothorn, Kurt Hornik, Achim Zeileis
   Journal of Computational and Graphical Statistics (2006-09) https://doi.org/d47hbq
   DOI: 10.1198/106186006x133933

4. **A General Coefficient of Similarity and Some of Its Properties**
   J. C. Gower
   Biometrics (1971-12) https://doi.org/dxtrt7
   DOI: 10.2307/2528823

5. **Ggplot2: elegant graphics for data analysis**
   Hadley Wickham
   Springer (2009)
   ISBN: 9780387981406

6. **partykit: A Modular Toolkit for Recursive Partytioning in R**
   Torsten Hothorn, Achim Zeileis
   Journal of Machine Learning Research (2015) http://jmlr.org/papers/v16/hothorn15a.html