rpms: An R Package for Modeling Survey Data with Regression Trees

Daniell Toth
U.S. Bureau of Labor Statistics

Abstract

In this article, we introduce the R package, **rpms** (Recursive Partitioning for Modeling Survey Data). The main function `rpms` fits a linear model to the data conditioning on the splits selected through the recursive partitioning algorithm. This application of recursive partitioning produces design consistent regression tree models of survey data with one-stage designs, accounting for any stratification and clustering as well as unequal probability of selection.

We examine the properties of the algorithm using samples from a simulated dataset and we provide a number examples using household level data from the U.S. Bureau of Labor Statistics' Consumer Expenditure survey. These applications demonstrate the easily interpretable structure and the ability of regression trees to handle large datasets with complex associations among the variables.

Keywords: machine learning, nonparametric, recursive partitioning, sample design.

1. Introduction

In this article we present the R (R Core Team 2016) package **rpms** (daniell toth 2017). This package provides a function `rpms` that returns an object of class "rpms", which is a representation of a regression tree model, as well as a number of functions that operate on objects of this class. A design-consistent regression tree is achieved by recursively partitioning the dataset and fitting the specified linear model on each node separately. The variable on which to split is selected at each stage using a nonparameteric test on the residuals and the splitting is ended using a stopping-rule based on this test. This leads to an algorithm that has unbiased variable selection and design consistent model parameters. The algorithm assumes a one-stage design that accounts for stratification and clustering as well as unequal probability of selection.

In Section 2 we will demonstrate the main function of the package, `rpms`, which returns an object of class "rpms" and the methods defined for this class. Section 2.4 contains an application of the package to U.S. Consumer Expenditure (CE) survey data.

2. Demonstrations

In this section we demonstrate the functions provided in the **rpms** package. In order to demonstrate the properties of the algorithm we simulate a population `simd` of clustered data.

2.1. The Simulated Data

There are 10000 observations of the random variables, $(y, x, va, vb, vc, vd, ve, vf)$, where x and y are continuous random variables and the v-variables are categorical. Each observation $(y_i, x_i, va_i, vb_i, vc_i, vd_i, ve_i, vf_i)$, is independent and identically distributed within a cluster, but the distributions of some of the random variables depend on the cluster.

For example, the values of vc are the same for every observation in a given cluster, and the values of x are correlated within the cluster, but independent between clusters. The dataset contains the label `id`, which is unique for each cluster. Figure 1 shows a plot of the variables y and x, where observations from two randomly chosen clusters are colored to show the homogeneity of y and x within clusters.

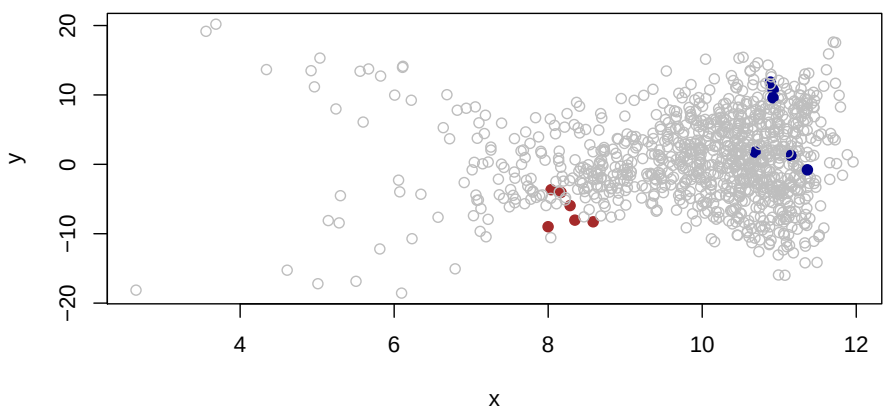


Figure 1: Plot of simulated population values, with variable y on the vertical-axis and x on the horizontal-axis. Two randomly chosen clusters are colored (blue and brown). This plot shows the homogeneity of y and x within clusters but does not seem to show a linear relationship between x and y.

Despite this scatter plot of the data not showing a linear relationship between x and y, the random variable y was simulated from a linear model. The i-th observation of variable y in cluster j is determined by

$$y_{ij} = \beta_{ij}(x_{ij} - 9) + \nu_j + \epsilon_{ij}, \tag{1}$$

where ν_j and ϵ_{ij} are mean zero noise terms, and β_{ij} is a function of the random variable va. The coefficient $\beta_{ij} = 3$ if the categorical variable $va_{ij} \in \{0, 1\}$ and $\beta_{ij} = -2$ if $va_{ij} \in \{2, 3\}$.

Figure 2 is the same plot with the points that have $va_{ij} \in \{0, 1\}$ colored blue and the points with $va_{ij} \in \{2, 3\}$ colored gold. Then the linear relationship is obvious.

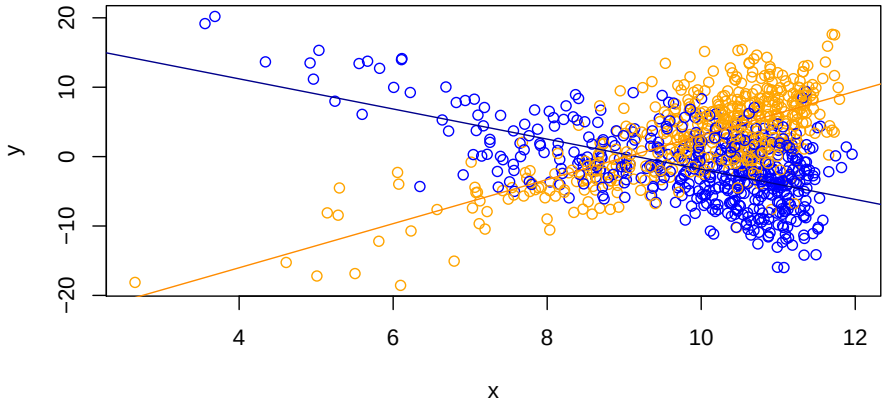


Figure 2: Plot of simulated population values, with variable y on the vertical-axis and x on the horizontal-axis. In this plot with the points with $va_{ij} \in \{0, 1\}$ colored blue and the points with $va \in \{2, 3\}$ colored gold, the linear relationship between x and y becomes obvious.

We will use this simulated dataset `simd` to demonstrate the `rpms` function and several functions provided in the package that operate on "rpms" objects. Since `rpms` is intended to allow inference about a population using a regression tree estimated from survey data, we will treat the dataset as the population and sample from it with different sampling designs to provide samples for use in the examples.

2.2. The `rpms` Function

The `rpms` function has two required arguments: `rp_equ` and `data`. The recursive partitioning is an R formula which identifies the dependent variable on the left-side of the "`~`" and specifies which variables the algorithm should consider for splitting the dataset on the right-side. Each variable is separated by "`+`" symbol and every variable listed in the `rp_equ` should be the name of either a numeric or categorical variable in the data.frame specified by the `data` argument.

The function `rpms` has an optional argument `e_equ` which is also an R formula. This argument is used to specify the linear equation the algorithm should fit to the dependent variable of each node of the tree. The dependent variable in `rp_equ` and `e_equ` must be the same. If y is the dependent variable and the `e_equ` is not given, by default, the function fits the linear equation $y = \beta$ to the data in each end-node. In this case, the parameter β is the mean of y given the observation is contained within that end-node.

Simple Random Sample

The following code fits a regression tree model for $E[y]$ using data from a simple random sample of size $n = 400$ from the population and assigns it to an object called `iid_tree`.

```
R> s0<-sample(N, n)
R> iid_tree <- rpms(rp_equ = y~va+vb+vc+ve+vf, data = simd[s0,])
```

```
R>
```

The result of the `rpms` call above is a binary tree with one split on the variable va shown in Figure 3. The algorithm chose to split only on the variable va the only variable included in the recursive partitioning equation to which the dependent variable y is related (Equation 1). According to the tree, using this one sample of size $n = 400$, observations with a value of $va \in \{0, 1\}$, the mean value of y is 3.19 and -1.76 for observations with $va \in \{2, 3\}$. The population means of y for the two domains are 2.93 and -1.66 respectively.

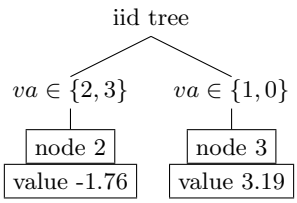


Figure 3: This is the tree fit from the simple random sample.

The rpms object

The object `iid_tree` is an R object of class "rpms", which is a list with 9 elements and "print" and "predict" methods defined for it. The internal elements of the object contain the necessary information to describe the tree model and allow the methods and other functions to operate on the object. Though the information is contained there, it was intended to be accessed through the more user friendly methods and functions.

The Print Method

The class "rpms" has a print method. The output of this method depends on whether the `e_equ` fits the mean or a linear equation with at least one independent variable. In the case of `iid_tree`, the estimating equation defaulted to fitting the mean at each end node, and so the output of print method is

```
R> iid_tree

RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf

Estimating Equation
y ~ 1

     Splits               Coefficients      SE
[1,]  1                   3.19184684906745  0.424498241381275
[2,]   va %in% c('2','3') -4.95123850066041 0.542267297586905
```

The first item displayed is the variables considered for partitioning and the second is the model fit at the end of each node. In this case there was no model given in the call to `rpms` so the model, $y = \beta$, is fit at the end of each node, so β is the mean of the dependent variable y in each node.

The last item displayed by the method is a linear representation of the fitted regression tree model. The first column shows the split and the next two give the coefficient and its design based standard error. The coefficient can be interpreted as the effect satisfying that split has

on the mean. For example by examining `iid_tree` for observation i begin with a estimate for $E[y_i]$ of 3.19. If the the observation satisfies the split condition, that is, if $va_i \in \{2, 3\}$, then add -4.95 to the estimate of $E[y_i]$. If the tree had more than one split, you would continue adding the coefficient of every split satisfied by the observation to the estimate of $E[y_i]$.
If we estimated a tree fitting the linear model $y = \beta_1 x + \beta_0 + \epsilon$ on each node, then the print method displays only the coefficients of the model and not the standard errors. For example, using the data from the same simple random sample,

```
R> iid_reg <- rpms(rp_equ = y~va+vb+vc+ve+vf, e_equ = y~x, data = simd[s0,])
R> iid_reg

RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf

Estimating Equation
y ~ x

     Splits
[1,]  1
[2,]   va %in% c('2','3')

    coefficients
node         1         x
   2  15.51409 -1.756959
   3 -24.60653  2.837154
```

The Predict Method

Once we have an rpms object we can use it to predict new values given values of the covariates used for the recursive partitioning and in the estimating equation. For example, suppose we have a dataframe

```
R> new

      x va vb vc vd ve vf
1 10.44  1  2  2  7  0  1
2 10.49  0  2  3  4  0  0
3 10.61  1  2  1 12  0  0
4 10.99  1  1  3  4  0  0
5  4.73  1  2  1  2  0  0
6 10.25  3  3  1  8  0  0
7 11.07  0  1  3 12  0  0
8 10.74  1  0  1  5  0  1
```

We can impute the 8 missing values for the random variable y as predicted by the regression tree model.

```
R> predict(iid_reg, newdata=new)

[1]   5.013356   5.155214   5.495672   6.573791 -11.186794  -2.494730
[7]   6.800763   5.864502

R> simd$y[pre1]

[1]  3.8127786 -0.1535147  9.5734612 -2.8995586 -9.2390248 -2.0460450
[7]  1.4893296  0.6775569

R>
```

Cluster Sample

Now instead of a simple random sample of our data we will consider a cluster sample. We take all the observations in a simple random sample of 20 clusters and use this data to illustrate the importance of accounting for clusters in the sample design.

```
R> cs0 <- which(simd$id %in% sample(unique(simd$id), 20))
```

Calling `rpms` below without identifying cluster labels and assigning it to `ctree1`, we get a tree model that splits not only on the variable *va*, but also on the variable *vc*.

```
R> ctree1 <- rpms(rp_equ = y~va+vb+vc+ve+vf, data = simd[cs0,])
R> ctree1

RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf

Estimating Equation
y ~ 1

     Splits
[1,]  1
[2,]   vc %in% c('3','1')
[3,]   vc %in% c('3','1') & va %in% c('2','3')
[4,]   vc %in%  c('4','0','2','5') & vc %in% c('4','2')
     Coefficients       SE
[1,] 3.19211856952504   1.33122409918345
[2,] -0.45360094288357  1.39138271605229
[3,] -4.3179932159343   0.598202094064419
[4,] -0.203916499835894 1.48849769357275
```

The algorithm chose to split on *vc* though, we know from Equation (1), that values of y do not depend on the values *vc*. This happens because the y values are correlated within a clusters and the value of *vc* are constant within a cluster. Since we only observe data from 20 clusters, it is likely that the algorithm finds partitions where y values are homogeneous using values of *vc* to split. Because the algorithm assumes that the values come from a sample of 400 independently selected observations the algorithm finds these splits to be significant. Accounting for the cluster sample design by specifying the cluster identification variable *id* in the call to `rpms`, results in a different tree model.

```
R> ctree2 <- rpms(rp_equ = y~va+vb+vc+ve+vf, data = simd[cs0,], clusters = ~id)
R> ctree2

RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf

Estimating Equation
y ~ 1

     Splits               Coefficients      SE
[1,]  1                   2.56735967893458  1.63061027821869
[2,]   va %in% c('2','3') -1.86796649483421 2.22432464833055
```

After identifying the clusters, the algorithm no longer finds splits on the variable *vc* significant and so the resulting model is very similar to the model that used simple random sample data and that captures the model used to generated the simulated data. The like `iid_tree`, `ctree2` only has the one split on *va*, however the parameter estimates of each end node are closer to

the population values for `iid_tree` than they are in `ctree2`. The model `ctree2` is estimated using data that contains much less information than 200 independent observations so the parameter estimates are more variable. This is reflected in the estimated standard deviations of the parameter estimates, which are three times as large for `ctree2` than the parameter estimates of `iid_tree`.

2.3. Functions for Visualizing Trees

Besides the print function, the `rpms` package contains two functions which help the user visualize the regression-tree model fit using the `rpms` function. The first, a function called `qtree`, we already saw an example of in Figure 3. In order to make that figure in LaTex we use a the `qtree` function and put the command

```
\usepackage{qtree}
```

in the preamble of our document.
The exact call to `qtree` in order to make the LaTeX code necessary to draw Figure 3 was:

```
R> qtree(iid_tree, title="iid tree", label="iidtree",
+        caption="This is the tree fit from the simple random sample.")
```

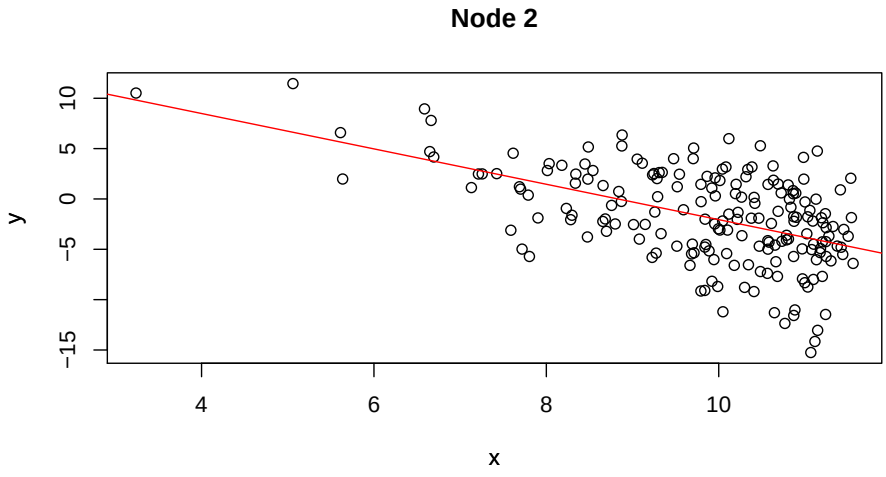
which produces the following output:

```
\begin{figure}[ht]
\centering
\footnotesize
\Tree [.{iid tree} [.{$va \in \{2,3\}$} {\fbox{node 2} \\ \fbox{value -1.76}} ][.{$va \in \{1,0\}$} {\fbox{node 3} \\ \fbox{value 3.19}} ]]
\caption{This is the tree fit from the simple random sample.}
\label{iidtree}
\end{figure}
```

This is code that can be directly put into a LaTeX document when the package "qtree" is being used.

The second function is `node_plot` which produces a graph of the data and the linear model fit to the data in the specified end_node of the tree. For example, the tree `iid_reg` had end_nodes 2 and 3, so we can look at the linear model fit at node 2 with the command:

```
R> node_plot(iid_reg, node = 2, data = simd[s0,])
```



2.4. CE Examples

In this section, we demonstrate some of the functions in the package with examples using household and expenditure data from the Consumer Expenditure Interview dataset for the first quarter of 2014. The dataset `CE` has 6483 observations of 61 variables collected from 70 clusters. This dataset is included (and automatically loaded) with the `rpms` package.

We will use the `rpms` function to explore the relationship between a number of household variables and the propensity of households, with income from earned wages, to contribute to a retirement plan. The percent of working households that contribute to a retirement account is estimated conditioning on a number of characteristics of the household.

We estimate the regression tree model both ignoring the cluster sample and then accounting for sample design using the design information contained in the dataset. Most of the splits obtained in the tree model when we ignore the clustering are removed once the clusters are accounted for. This illustrates that the clustering of many of the household variables reduces the information obtained from the survey. This leads to less certainty concerning differences in propensity between households with different characteristics observed in the data.

```
R> CE$saver <- ifelse(CE$FINDRETX>0, 1, 0)
R> rate_tree0 <-
+    rpms(rp_equ = saver~FAM_SIZE+FINCBTAX+NO_EARNR+PERSOT64+CUTENURE+VEHQ+REGION,
+                  weights = ~FINLWT21, data = CE[workers,], pval=.01)
R> rate_tree1 <-
+    rpms(rp_equ = saver~FAM_SIZE+FINCBTAX+NO_EARNR+PERSOT64+CUTENURE+VEHQ+REGION,
+                  weights = ~FINLWT21, clusters = ~CID, data = CE[workers,], pval=.01)
R>
```

In each call to the function `rpms` an object is assigned the tree model with the estimated percentage of working households that contribute a retirement account at each end-node. In the first model, `rate_tree0` is the resulting tree which accounts for the unequal probability of selection by identifying the sample weights `FINLWT21` but ignores the clustered sample-design.

```
R> rate_tree0

RPMS Recursive Partitioning Equation
saver ~ FAM_SIZE + FINCBTAX + NO_EARNR + PERSOT64 + CUTENURE +
    VEHQ + REGION

Estimating Equation
saver ~ 1

      Splits
 [1,]  1
 [2,]   NO_EARNR <= 1
 [3,]   NO_EARNR <= 1 & CUTENURE %in% c('2','1')
 [4,]   NO_EARNR <= 1 & CUTENURE %in% c('2','1') & FINCBTAX <= 62180
 [5,]   NO_EARNR <= 1 & CUTENURE %in% c('2','1') & FINCBTAX <= 62180 & FINCBTAX <= 42005
 [6,]   NO_EARNR <= 1 & CUTENURE %in%  c('4','5','6') & FINCBTAX <= 35400
 [7,]   NO_EARNR > 1 & FINCBTAX <= 93400
 [8,]   NO_EARNR > 1 & FINCBTAX <= 93400 & CUTENURE %in% c('1')
 [9,]   NO_EARNR > 1 & FINCBTAX <= 93400 & CUTENURE %in% c('1') & FINCBTAX <= 68145
[10,]   NO_EARNR > 1 & FINCBTAX <= 93400 & CUTENURE %in%  c('4','2','5') & FAM_SIZE <= 2
[11,]   NO_EARNR > 1 & FINCBTAX > 93400 & FINCBTAX <= 142000
      Coefficients        SE
 [1,] 0.267191084267725   0.000168095257831745
 [2,] -0.164913163265835  0.000207281404569445
 [3,] 0.0611796087796629  0.000180811820393956
```

```
 [4,] -0.0527486516669274 0.000199391178164796
 [5,] -0.0607434678993177 0.000167306590520415
 [6,] -0.0875101654042774 0.000126077029899887
 [7,] -0.2434144364817    0.00017559944370445
 [8,] 0.113664872188168   0.000168741627976171
 [9,] -0.0493068600062255 0.000197689837557363
[10,] 0.0439771080826453  0.000115889209161041
[11,] -0.0757584535962249 0.000218364371482939

R> rate_tree1

RPMS Recursive Partitioning Equation
saver ~ FAM_SIZE + FINCBTAX + NO_EARNR + PERSOT64 + CUTENURE +
    VEHQ + REGION

Estimating Equation
saver ~ 1

     Splits              Coefficients       SE
[1,]  1                  0.182420286093682  8.48303949589336e-05
[2,]   FINCBTAX <= 71511 -0.127244995276125 8.06872366021757e-05
```

In the second model, `rate_tree1` is the resulting which accounts for the cluster sample-design with unequal probability of selection by adding the argument `clusters = ~CID`. This informs the algorithm that observations with equal values of `CID` are from the same cluster. The resulting tree has only the first split on household income of `rate_tree0`. Both models used a p-value of .05, and were unchanged using a value of .01, which suggests that we would choose `rate_tree1` to make inference, if we were concerned about overfitting the data. However, if we are more interested in getting accurate predictions and not as worried about overfitting, we might use `rate_tree0`.

References

daniell toth (2017). *rpms: Recursive Partitioning for Modeling Survey Data*. R package version 0.2.0.

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

Affiliation:

Daniell Toth
Office of Survey Methods Research
Senior Research Mathematical Statistician
U.S. Bureau of Labor Statistics
2 Massachusetts Avenue NE// Washington, DC 20212
Telephone: +1/202/691-7380 E-mail: toth.daniell@bls.gov