Recursive Partitioning for Modelling Survey data

An Introduction to the R package: rpms

Daniell Toth

U.S. Bureau of Labor Statistics

Content represents the opinion of the authors only.

Talk Outline

- Brief introduction of recursive partitioning
- Description of simulated data used for demonstrations
- Demonstrate some of the functionality of the rpms package:
    - tree regression with rpms
    - including sample design information
    - examples using provided functions
- Example Using Consumer Expenditure Data

Recursive Partitioning

full dataset

θ

Population has some unknown parameters θ
that we wish to estimate

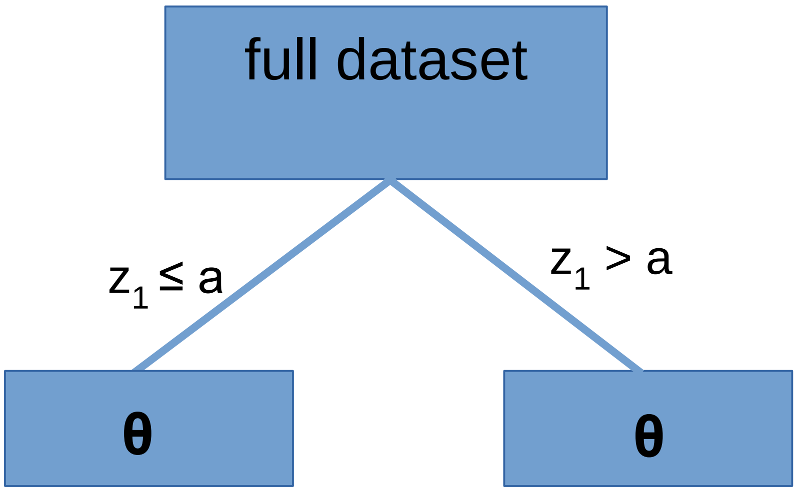
Recursive Partitioning

full dataset

θ

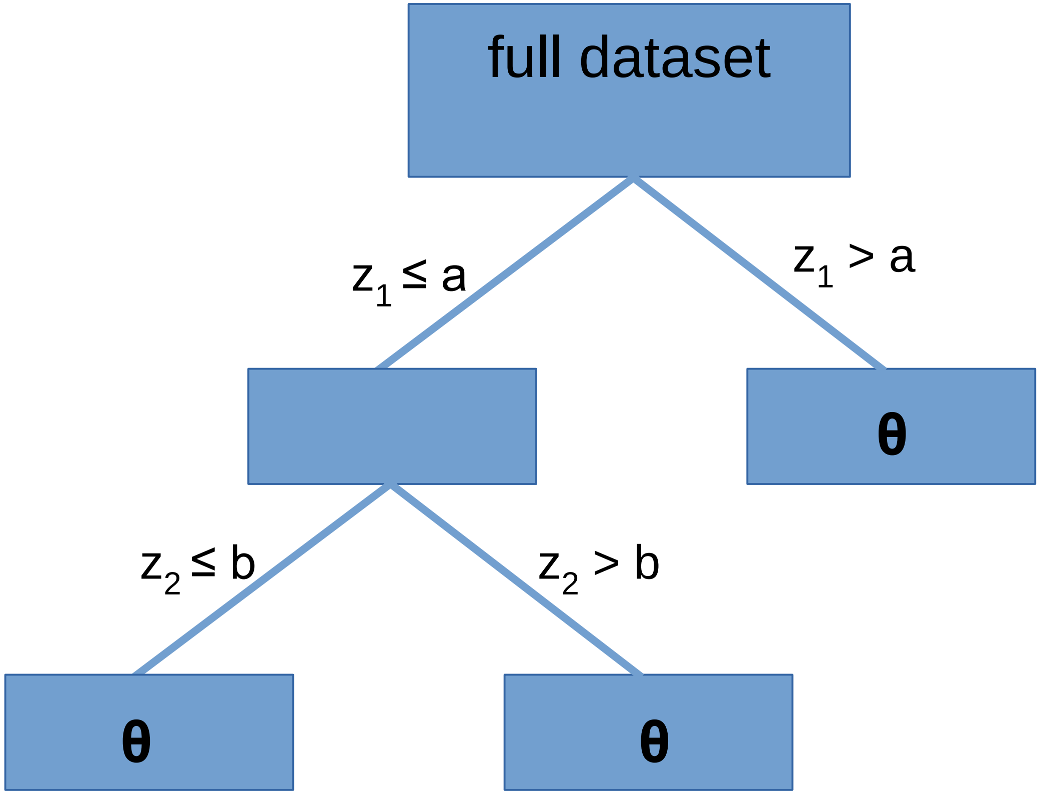
θ is often the mean, but could be other model parameters such as variance, proportion or coefficients of a linear model

Recursive Partitioning

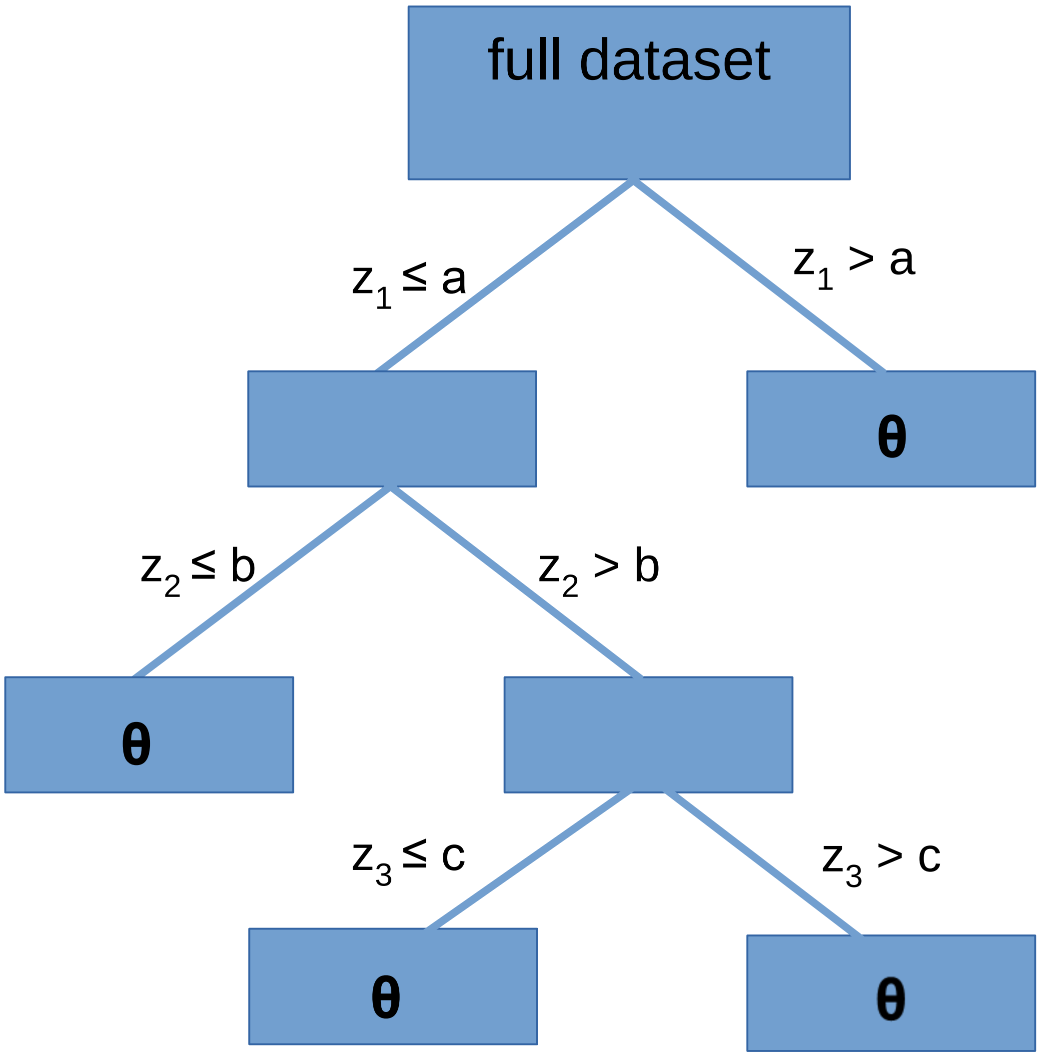


θ could be very different for different sub-domains of the population

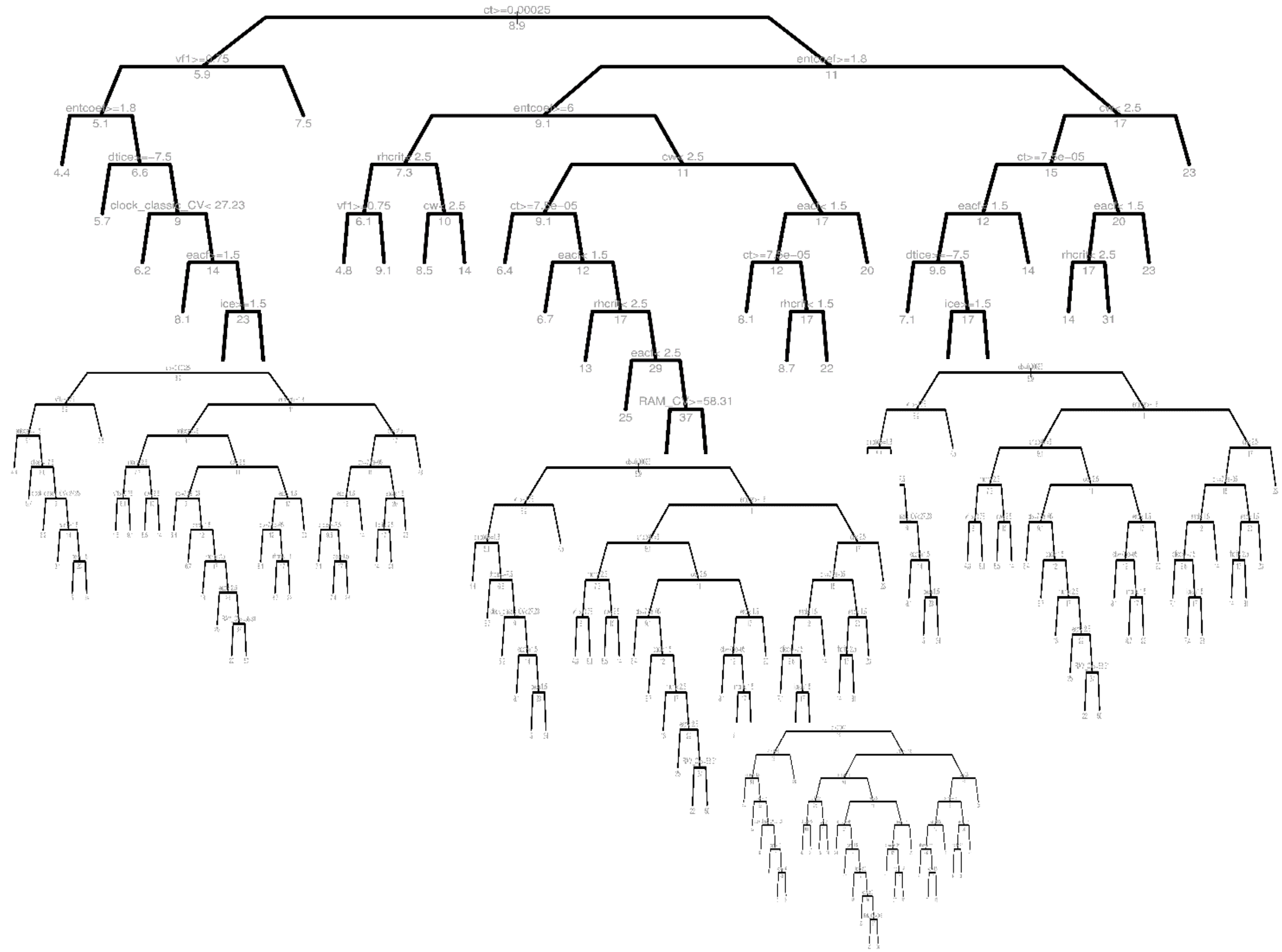
Recursive Partitioning



Recursive Partitioning



Ad infinitum



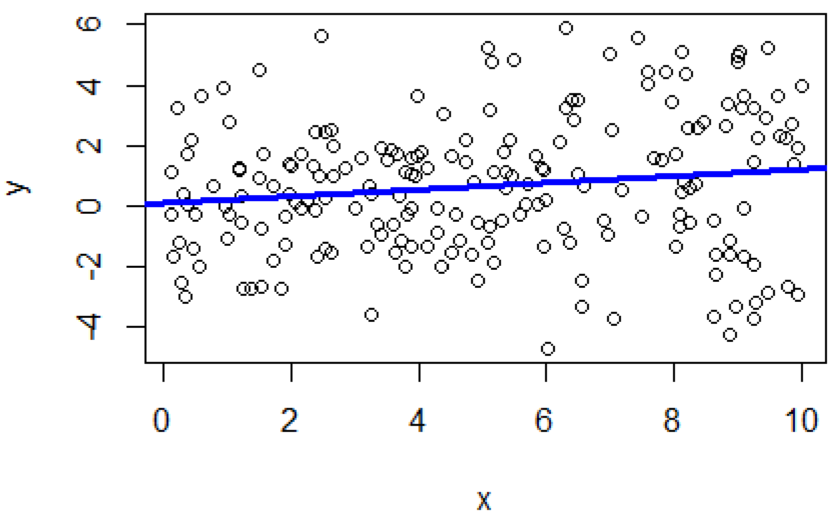
The Problem

We wish to understand the relationship between a variable Y and number of other available variables in **X** and **Z**

- (Y, **X**, **Z**) come from a complex sample
- Y ~ **X** linear relationship (parametric)
- $\mathbf{X}^T\beta \sim \mathbf{Z}$ unknown and complicated (nonparametric)
- **Z** contains many variables (large p);
- $\mathbf{X}^t\beta$ depends on subset of **Z** and interactions (variable selection)
- Some variables in **Z** maybe collinear

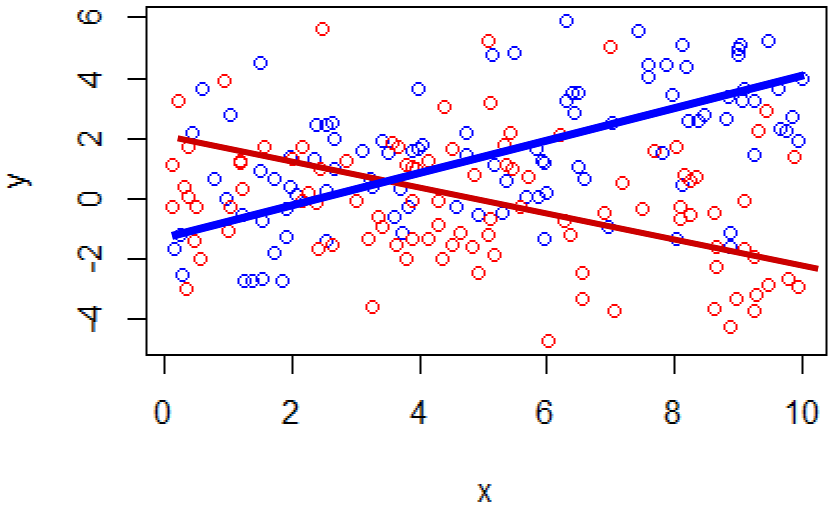
Regression Example

full dataset
$y=x^T\beta$



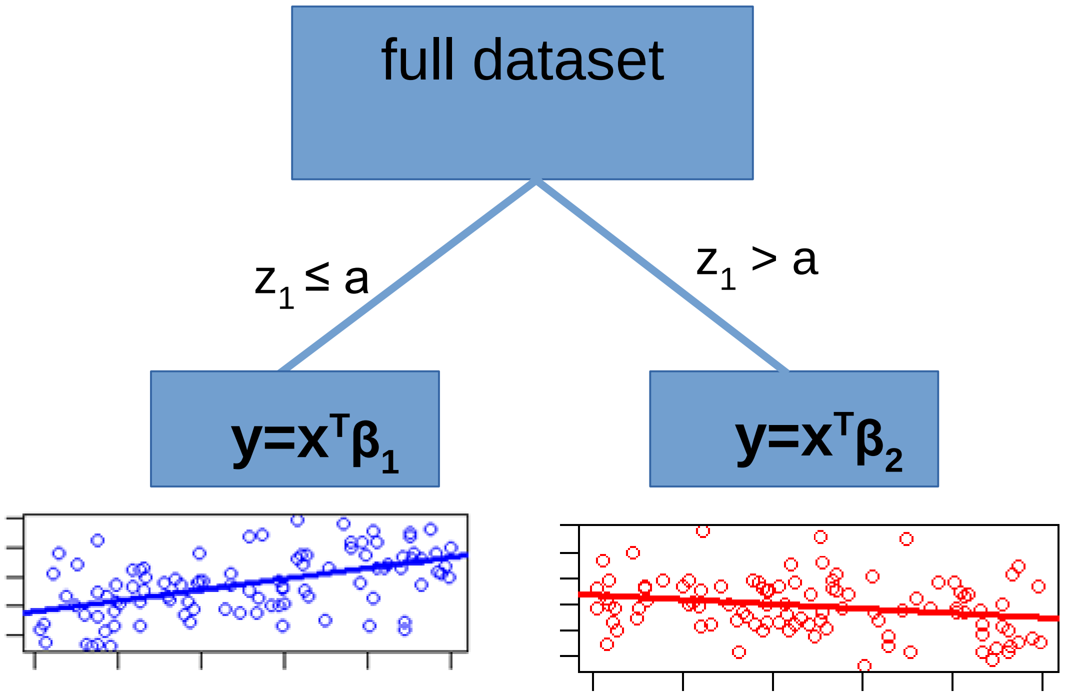
Regression Example

full dataset
$y=x^T\beta$



$z_1 \leq a$ $z_1 > a$

Regression Example



CRAN – Package rpms

rpms: Recursive Partitioning for Modeling Survey Data

Fits a linear model to survey data in each node obtained by recursively partitioning the data. The splitting variables and splits selected are obtained using a procedure which adjusts for complex sample design features used to obtain the data. Likewise the model fitting algorithm produces design-consistent coefficients to the least squares linear model between the dependent and independent variables. The first stage of the design is accounted for in the provided variance estimates. The main function returns the resulting binary tree with the linear model fit at every end-node. The package provides a number of functions and methods for these trees.

Version:	0.2.0
Depends:	R (≥ 2.10)
Imports:	Rcpp (≥ 0.12.3)
LinkingTo:	Rcpp, RcppArmadillo
Published:	2017-02-04
Author:	daniell toth [aut, cre]
Maintainer:	daniell toth <danielltoth at yahoo.com>
License:	CC0
NeedsCompilation:	yes
In views:	OfficialStatistics
CRAN checks:	rpms results

Downloads:

Reference manual:	rpms.pdf
Vignettes:	An Introduction to rpms
Package source:	rpms_0.2.0.tar.gz
Windows binaries:	r-devel: rpms_0.2.0.zip, r-release: rpms_0.2.0.zip, r-oldrel: rpms_0.2.0.zip
OS X Mavericks binaries:	r-release: rpms_0.2.0.tgz, r-oldrel: rpms_0.2.0.tgz
Old sources:	rpms archive

Linking:

Please use the canonical form `https://CRAN.R-project.org/package=rpms` to link to this page.

The rpms package

```
>library(rpms)
```

Package provides a number of functions designed to help model and analyze survey data using design-consistent regression trees.

rpms - returns rpms object
methods:
    print, predict

other available functions:
    node_plot, qtree, in_node

The Simulated Data

We simulate a dataset of 10,000 observations simd

$y_{ij} = f(x_{ij,}\ \mathbf{v}_{ij}) + \eta_j + \varepsilon_{ij}$

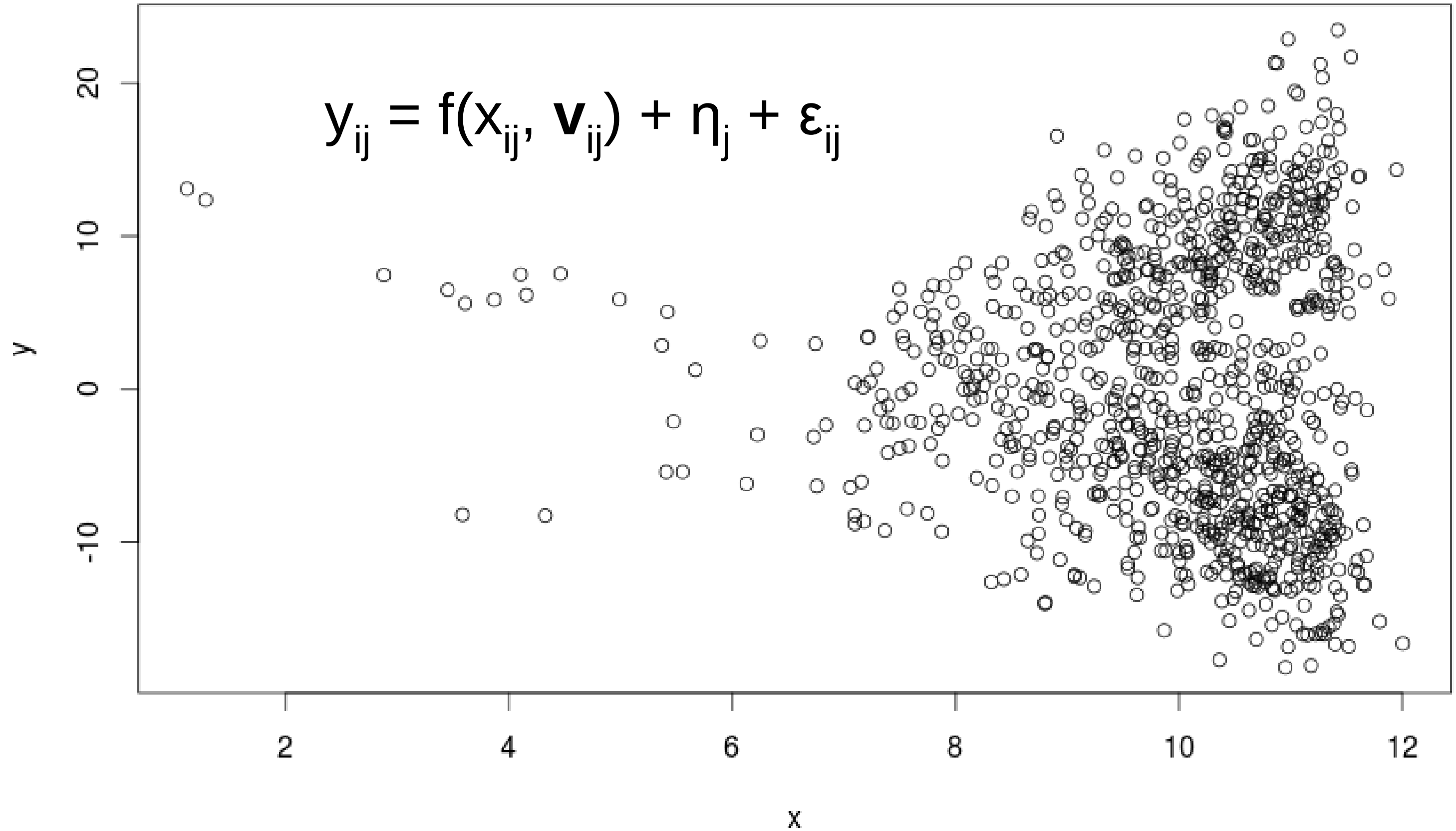
x = continuous variable

$v_{a,}\ v_{b,}\ \ldots\ v_f$ = categorical variables

v_a is the only variable that f depends on

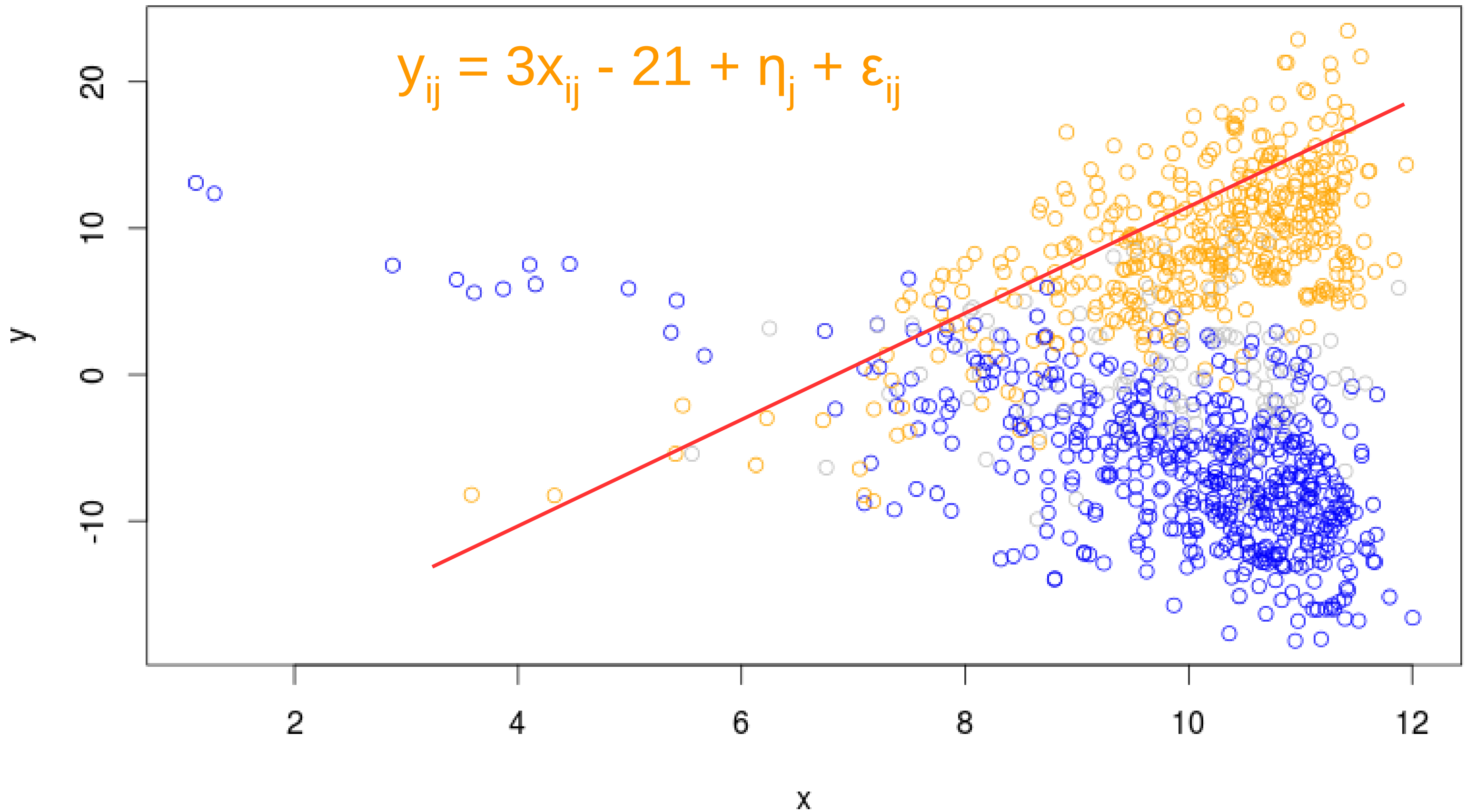
The Simulated Data

We simulate a dataset of 10,000 observations simd



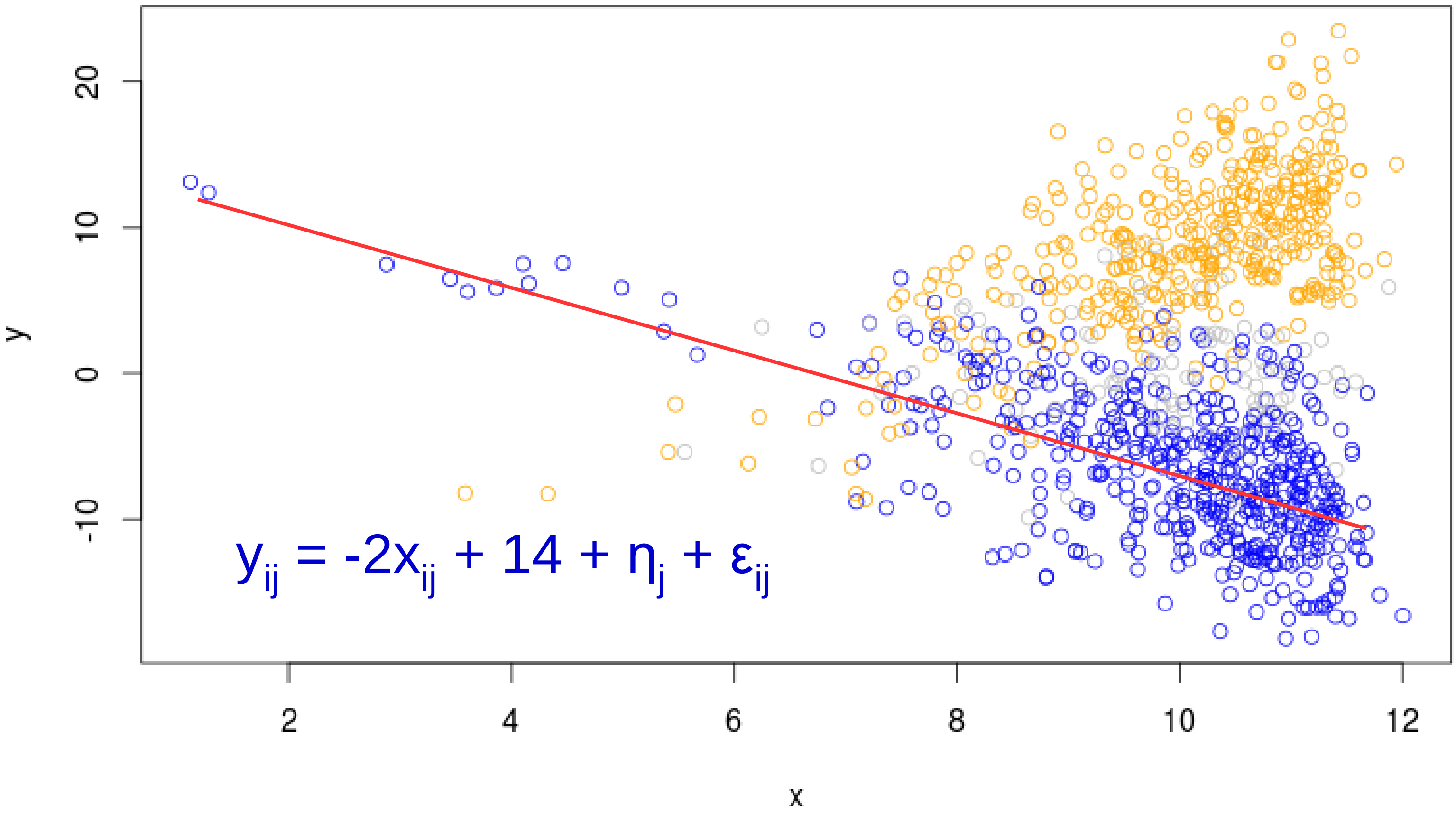
The Simulated Data

If $v_a = \{0,1\}$



The Simulated Data

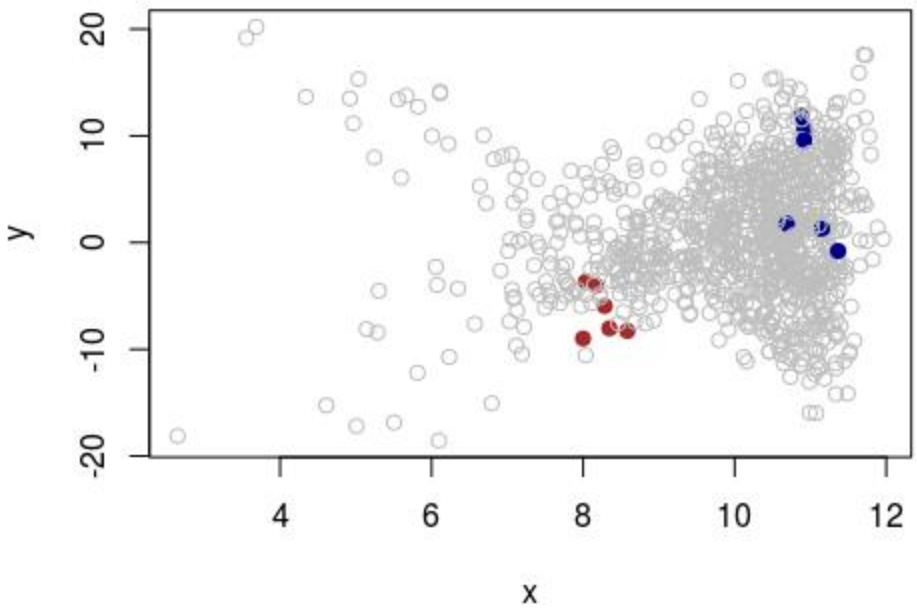
If $v_a \in \{2, 3\}$



The Simulated Data

We simulate a dataset of 10,000 observations simd

$$y_{ij} = f(x_{ij}, \mathbf{v}_{ij}) + \eta_j + \varepsilon_{ij}$$



Basic rpms Call

The function rpms only requires two things:

rp_equ	names the variables to potentially split on
data	data.frame object containing required variables

```
>r1<-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd)
```

Basic rpms Call

The function rpms only requires two things:

rp_equ	names the variables to potentially split on
data	data.frame object containing required variables

R formula

```
>iid <-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd[s,])
```

Basic rpms Call

The function rpms only requires two things:

rp_equ	names the variables to potentially split on
data	data.frame object containing required variables

R formula

```
>iid <-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd[s,])
```

data set containing:
  1.) splitting variables,
  2.) model variables,

may contain:
  3.) design variables

Basic rpms Call

The function rpms only requires two things:

rp_equ	names the variables to potentially split on
data	data.frame object containing required variables

```
>iid <-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd[s,])
```

splitting variables
seperated by +

Basic rpms Call

The function rpms only requires two things:

rp_equ	names the variables to potentially split on
data	data.frame object containing required variables

```
>iid <-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd[s,])
```

R assignment operator

iid is an rpms object

print.rpms

>print(iid)   same as >iid

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                    Coefficients      SE
[1,]  1                          3.1918        0.4245
[2,]   va %in% c('2','3')       -4.9512        0.5423
```

print.rpms

>print(iid)   same as >iid

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1
```

no e_eq
fits mean by
default

```
     Splits                      Coefficients        SE
[1,]  1                            3.1918          0.4245
[2,]   va %in% c('2','3')         -4.9512          0.5423
```

print.rpms

>print(iid)   same as >iid

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                        Coefficients       SE
[1,]  1                              3.1918          0.4245
[2,]   va %in% c('2','3')           -4.9512          0.5423
```

Linear form can be useful (Phipps & Toth 2012)

Population Parameters

rpms model on srs n=400

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                        Coefficients       SE
[1,]  1                              3.1918           0.4245
[2,]   va %in% c('2','3')          -4.9512           0.5423
```

rpms model on population

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf
Estimating Equation
y ~ 1
     Splits                        Coefficients       SE
[1,]  1                              2.9317         0.2765
[2,]   va %in% c('2','3')          -4.5939         0.3354
```

Population Parameters

rpms model on srs n=400

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                        Coefficients       SE
[1,]  1                              3.1918           0.4245
[2,]   va %in% c('2','3')           -4.9512           0.5423
```

rpms model on population

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf
Estimating Equation
y ~ 1
     Splits                        Coefficients       SE
[1,]  1                              2.9317         0.2765
[2,]   va %in% c('2','3')           -4.5939         0.3354
```

same split

Population Parameters

rpms model on srs n=400

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                     Coefficients      SE
[1,]  1                           3.1918          0.4245
[2,]   va %in% c('2','3')       -4.9512           0.5423
```

rpms model on population

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf
Estimating Equation
y ~ 1
     Splits                     Coefficients      SE
[1,]  1                           2.9317        0.2765
[2,]   va %in% c('2','3')       -4.5939         0.3354
```

similar coefficients

Population Parameters

rpms model on srs n=400

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                      Coefficients       SE
[1,]  1                            3.1918           0.4245
[2,]   va %in% c('2','3')         -4.9512           0.5423
```

rpms model on population

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf
Estimating Equation
y ~ 1
     Splits                      Coefficients       SE
[1,]  1                            2.9317         0.2765
[2,]   va %in% c('2','3')         -4.5939         0.3354
```

higher standard errors

qtree

Once we have an rpms object we can include its tree structure as a figure in a Latex paper or Sweave document

```
>qtree(iid)
```

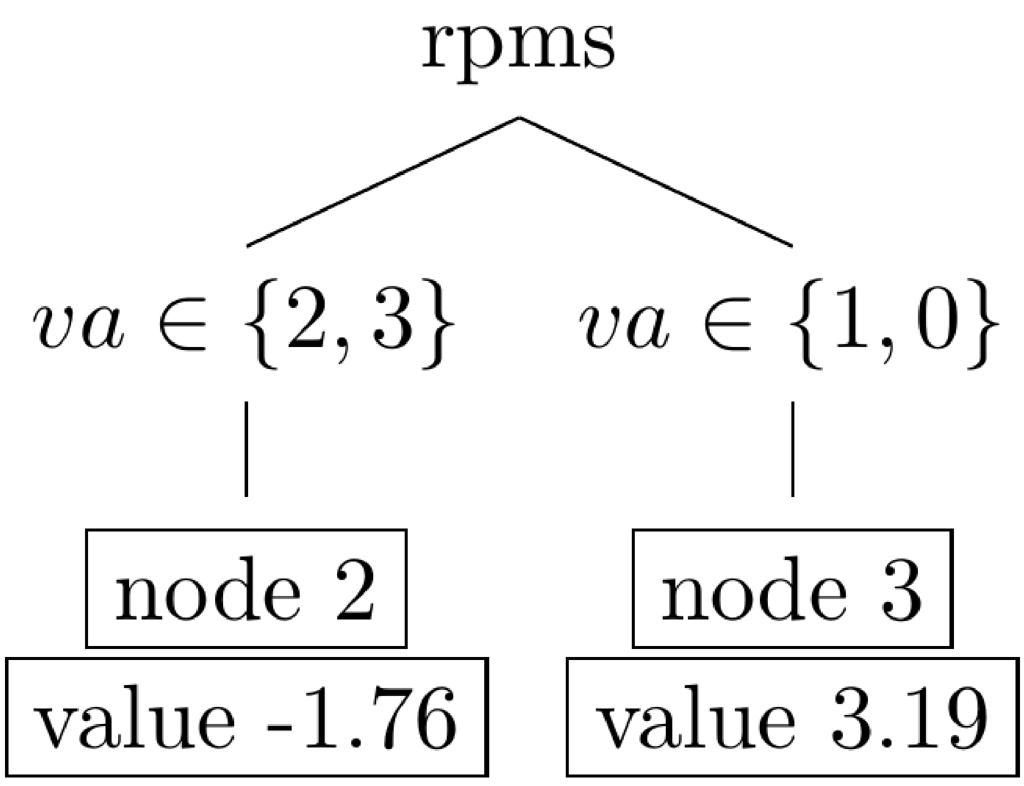
qtree

Once we have an rpms object we can include its tree structure as a figure in a Latex paper or Sweave document

>qtree(iid)

```
\begin{figure}[ht]
\centering
\begin{tikzpicture}[scale=1, ]
\tikzset{every tree node/.style={align=center,anchor=north}}
\Tree [.{rpms} [.{$va \in \{2,3\}$} {\fbox{node 2} \\ \fbox{value -1.76}}
][.{$va \in \{1,0\}$}
{\fbox{node 3} \\ \fbox{value 3.19}} ]]
\end{tikzpicture}
\caption{}
\end{figure}
```

qtree



Other options in qtree

qtree takes several optional parameters:

- title = character string for naming root node; default="rpms“
- label = string for LaTex figure label; used referencing figure,
- caption = string caption for figure, defaults to blank
- scale= number (0, ∞); changes relative size of figure; default=1

Other options in qtree

qtree takes several optional parameters:

- title = character string for naming root node; default="rpms“
- label = string for LaTex figure label; used referencing figure,
- caption = string caption for figure, defaults to blank
- scale= number (0, ∞); changes relative size of figure; default=1

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

qtree

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

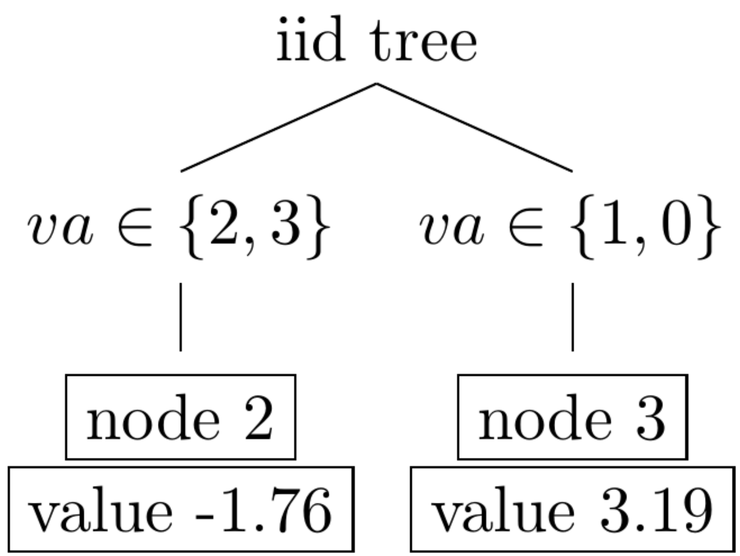


Figure 4: This is the tree fit from the simple random sample.

Tree Regression

Instead of estimating the mean of each node, we estimate the parameters to the equation $y = \beta x + \alpha$

```
>rx1<-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd
           e_equ=y~x)
```

Tree Regression

Instead of estimating the mean of each node, we estimate the parameters to the equation $y = \beta x + \alpha$

```
>rx1<-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd
           e_equ=y~x)
```

estimating equation

Tree Regression

Instead of estimating the mean of each node, we estimate the parameters to the equation $y = \beta x + \alpha$

```
>rx1<-rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd
              e_equ=y~x)
```

estimating equation

dependent variable must be the same for rp_equ and e_equ

Print Changes

>rx1

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf
Estimating Equation
y ~ x          <-- e_eq

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

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

Print Changes

>rx1

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + 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
```

α at node 2

Print Changes

>rx1

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + 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
```

α at node 2

β at node 2

node_plot

This lets you see each node with data and the fitted line with respect to chosen variable

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + 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
```

>node_plot(rx1, node=2, simd)

node_plot

This lets you see each node with data and the fitted line with respect to chosen variable

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + 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
```

rpms object

data

```
>node_plot(rx1, node=2, simd)
```

node_plot

This lets you see each node with data and the fitted line with respect to chosen variable

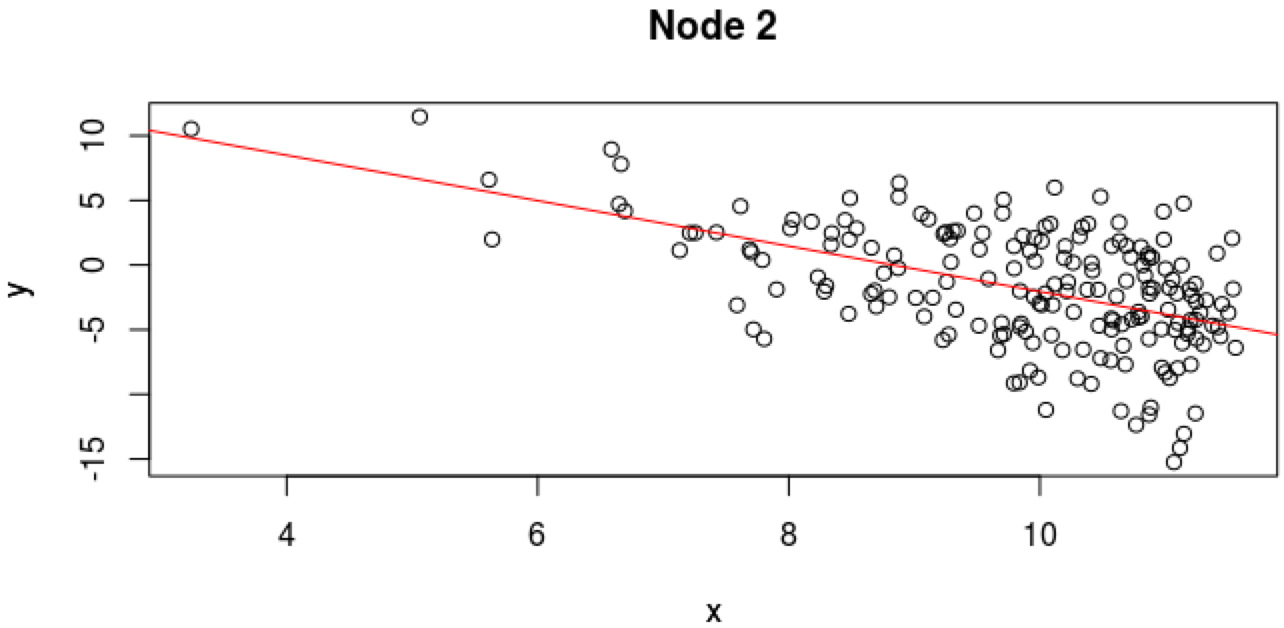
```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf
Estimating Equation
y ~ x

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

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

Parameter variable defaults to the first variable of e_eq

>node_plot(rx1, node=2, simd, variable = x)

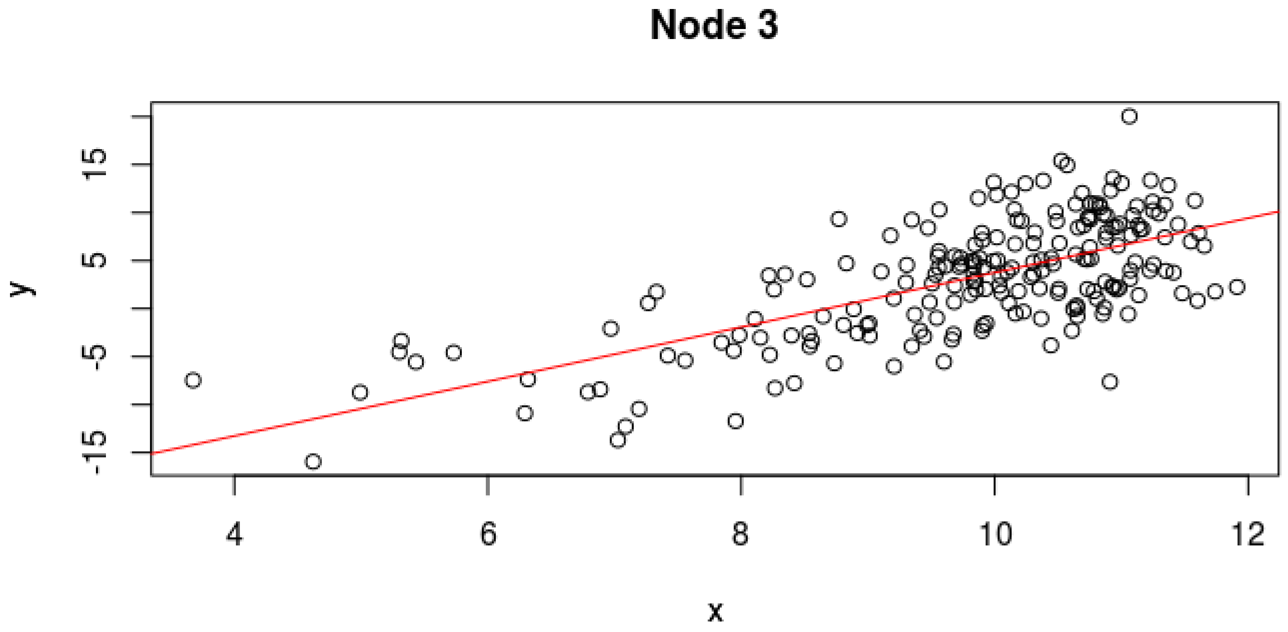
node 2

```
>node_plot(rx1, node=2, simd)
```

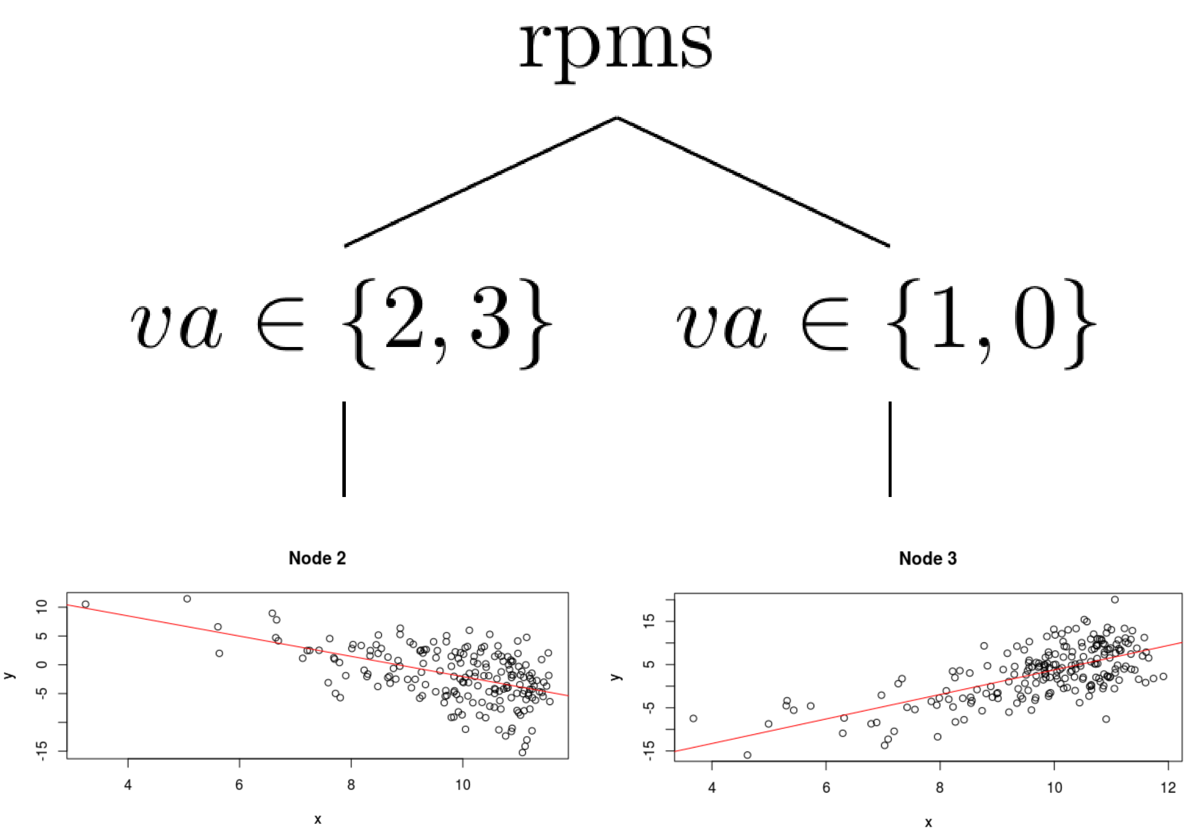


node 3

>node_plot(rx1, node=3, simd)



rx1



Data from a Complex Sample

The 10,000 observations simd were constructed by simulating 500 clusters with 20 observations each.

$y_{ij} = f(x_{ij}) + \eta_j + \varepsilon_{ij}$

x = continuous variable

$v_{a,}\ v_{b,}\ \dots v_f$ = categorical variables

Data from a Complex Sample

The 10,000 observations simd we constructed by simulating 500 clusters with 20 observations each.

$N(0, \sigma_c)$ same for each observation in cluster

$y_{ij} = f(x_{ij}) + \eta_j + \varepsilon_{ij}$

x = continuous variable

$v_{a,}\ v_{b,} \ldots v_f$ = categorical variables

Data from a Complex Sample

The 10,000 observations simd we constructed by simulating 500 clusters with 20 observations each.

$N(0, \sigma_c)$ same for each observation in cluster

$y_{ij} = f(x_{ij}) + \eta_j + \varepsilon_{ij}$

$x_j + e_{ij}$
more homogeneous within cluster

x = continuous variable

$v_{a,}\ v_{b,}\ \ldots\ v_f$ = categorical variables

Data from a Complex Sample

The 10,000 observations simd we constructed by simulating 500 clusters with 20 observations each.

$N(0, \sigma_c)$ same for each observation in cluster

$y_{ij} = f(x_{ij}) + \eta_j + \varepsilon_{ij}$

x = continuous variable

$x_j + e_{ij}$
more homogeneous within cluster

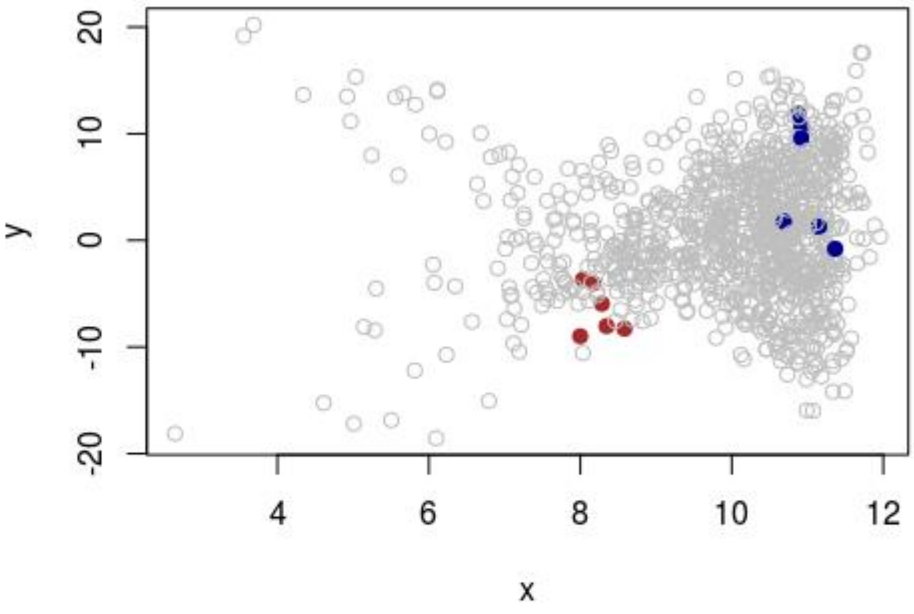
$v_{a,}\ v_{b,}\ \ldots v_f$ = categorical variables

v_c

same for every observation in cluster

Cluster Sample

SRS of clusters



Ignoring the Design

Ignoring the sample design when building the regression trees is a bad idea

>rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd)

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

Estimating Equation
y ~ 1

     Splits                                                        Coefficients       SE
[1,]  1                                                            -2.894254341  0.659906377
[2,]   va %in% c('1')                                              13.127085821  0.817361236
[3,]   va %in% c('1') & vc %in% c('4','2')                         -5.702057150  0.948519947
[4,]   va %in%  c('3','2','0') & va %in% c('2')                    -1.819929357  0.890137832
[5,]   va %in%  c('3','2','0') & va %in% c('2') & vc %in% c('4','1') -1.982788586  0.746303072
```

Ignoring the Design

Ignoring the sample design when building the regression trees is a bad idea

>rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd)

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

Estimating Equation
y ~ 1

     Splits                                                            Coefficients      SE
[1,]  1                                                                -2.894254341  0.659906377
[2,]   va %in% c('1')                                                  13.127085821  0.817361236
[3,]   va %in% c('1') & vc %in% c('4','2')                             -5.702057150  0.948519947
[4,]   va %in%  c('3','2','0') & va %in% c('2')                        -1.819929357  0.890137832
[5,]   va %in%  c('3','2','0') & va %in% c('2') & vc %in% c('4','1')   -1.982788586  0.746303072
```

variable vc included in model

Ignoring the Design

Ignoring the sample design when building the regression trees is a bad idea

>rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd)

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

Estimating Equation
y ~ 1

     Splits                                                         Coefficients      SE
[1,]  1                                                             -2.894254341  0.659906377
[2,]   va %in% c('1')                                               13.127085821  0.817361236
[3,]   va %in% c('1') & vc %in% c('4','2')                          -5.702057150  0.948519947
[4,]   va %in%  c('3','2','0') & va %in% c('2')                     -1.819929357  0.890137832
[5,]   va %in%  c('3','2','0') & va %in% c('2') & vc %in% c('4','1') -1.982788586  0.746303072
```

variable vc included in model

including psu labels ids

>rpms(rp_equ=y~va+vb+vc+vd+ve+vf, data=simd,
clusters=~ids)

Regression Tree with Cluster Design

accounting for clusters could change more than standard errors

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

Estimating Equation
Y ~ 1

     Splits                        Coefficients       SE
[1,]  1                              2.56736       1.63061
[2,]   va %in% c('2','3')           -1.86798       2.22432
```

Cluster Design vs iid

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

Estimating Equation
Y ~ 1

      Splits                     Coefficients      SE
[1,]  1                             2.56736      1.63061
[2,]   va %in% c('2','3')        -1.86798      2.22432
```

correct tree model

> iid

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

      Splits                     Coefficients      SE
[1,]  1                             3.1918       0.4245
[2,]   va %in% c('2','3')        -4.9512       0.5423
```

Cluster Design vs iid

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

Estimating Equation
Y ~ 1

      Splits                    Coefficients      SE
[1,]   1                          2.56736       1.63061
[2,]    va %in% c('2','3')       -1.86798       2.22432
```

> iid

```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

      Splits                    Coefficients      SE
[1,]   1                          3.1918        0.4245
[2,]    va %in% c('2','3')       -4.9512        0.5423
```

coefficients aren't as accurate

Cluster Design vs iid

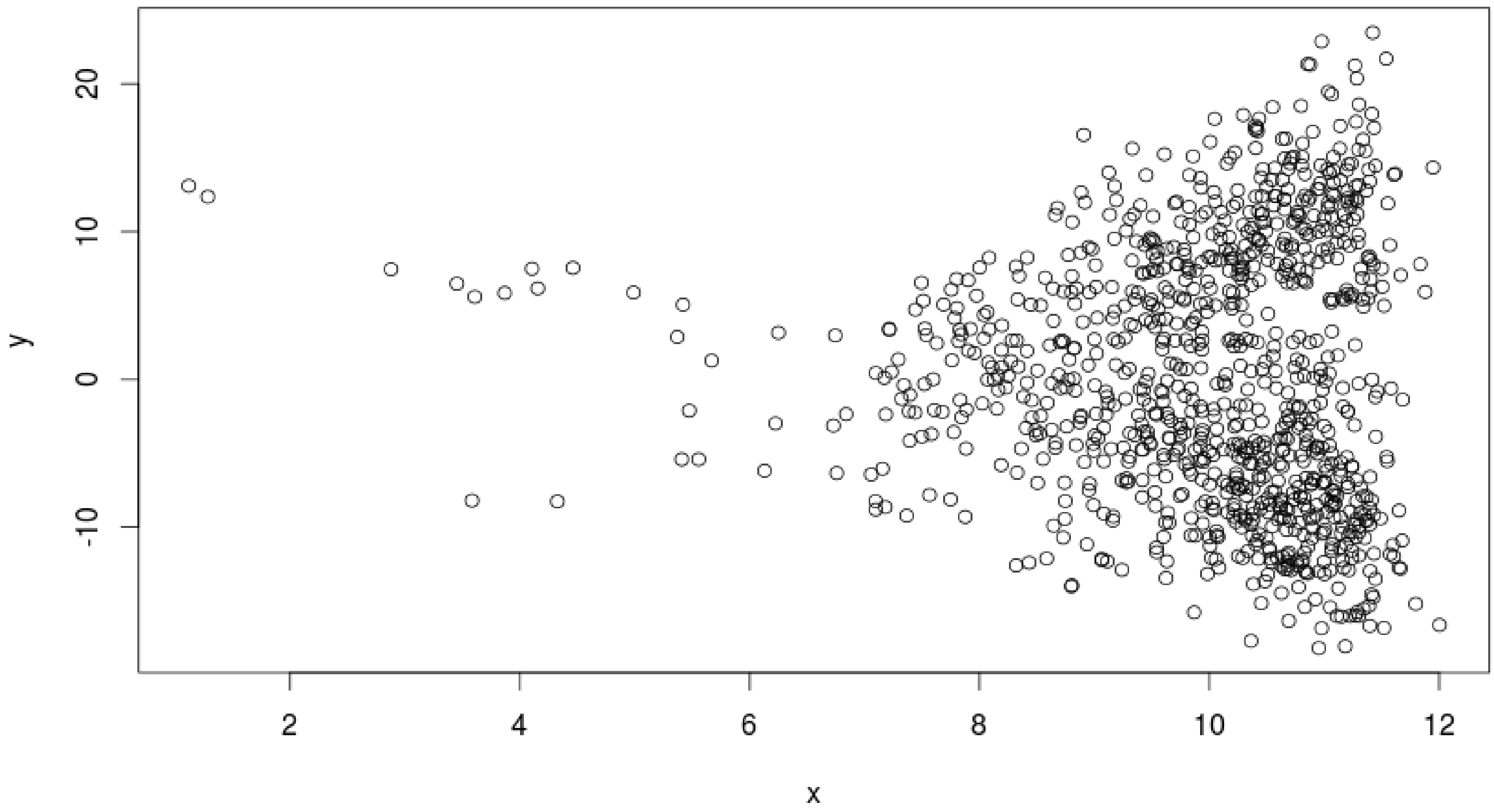
```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + vd + ve + vf

Estimating Equation
Y ~ 1

     Splits                    Coefficients      SE
[1,]  1                          2.56736       1.63061
[2,]   va %in% c('2','3')       -1.86798       2.22432
```

> iid

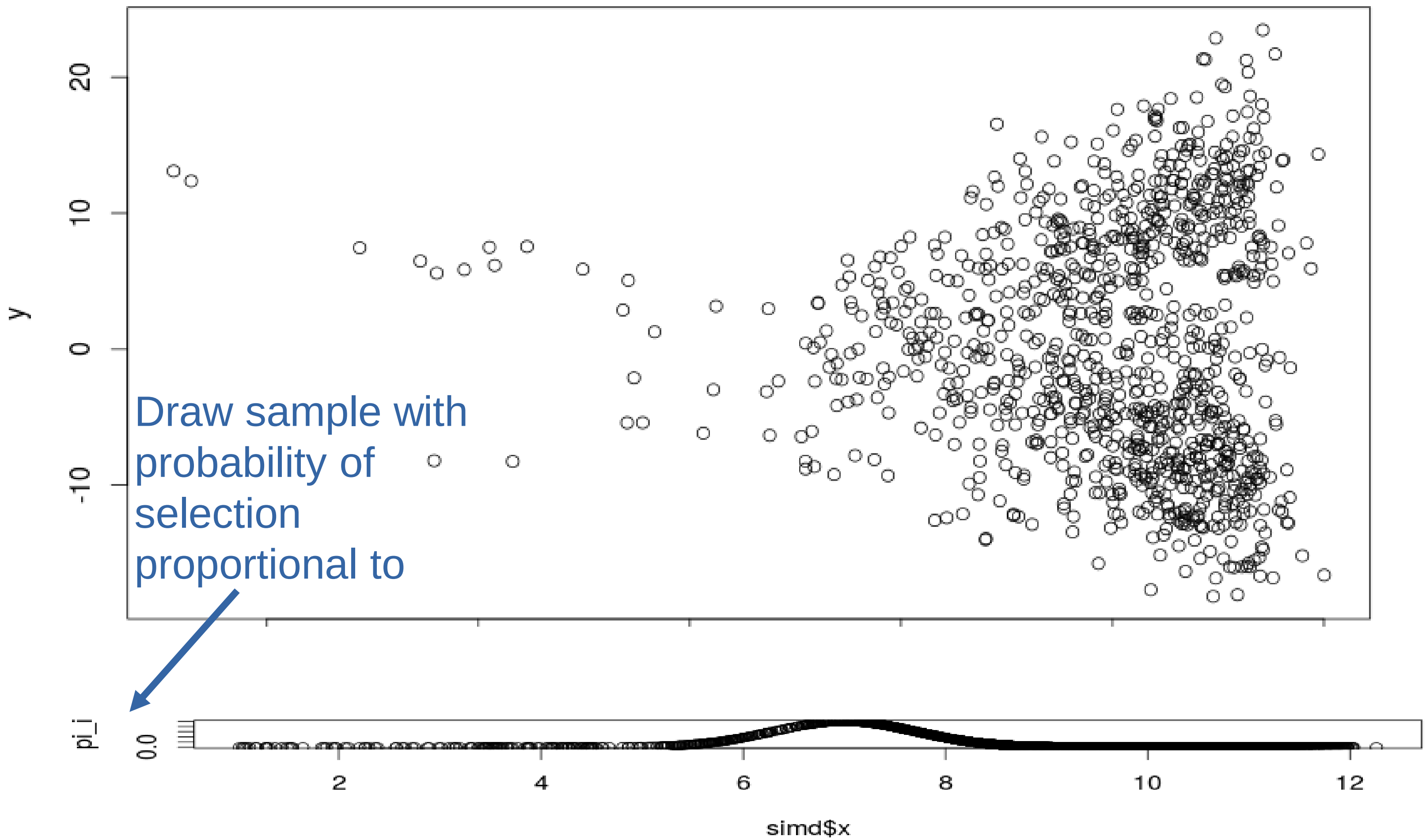
```
RPMS Recursive Partitioning Equation
y ~ va + vb + vc + ve + vf
Estimating Equation
y ~ 1

     Splits                    Coefficients      SE
[1,]  1                          3.1918        0.4245
[2,]   va %in% c('2','3')       -4.9512        0.5423
```

standard errors of coefficients DID increase

Unequal Probability of Selection

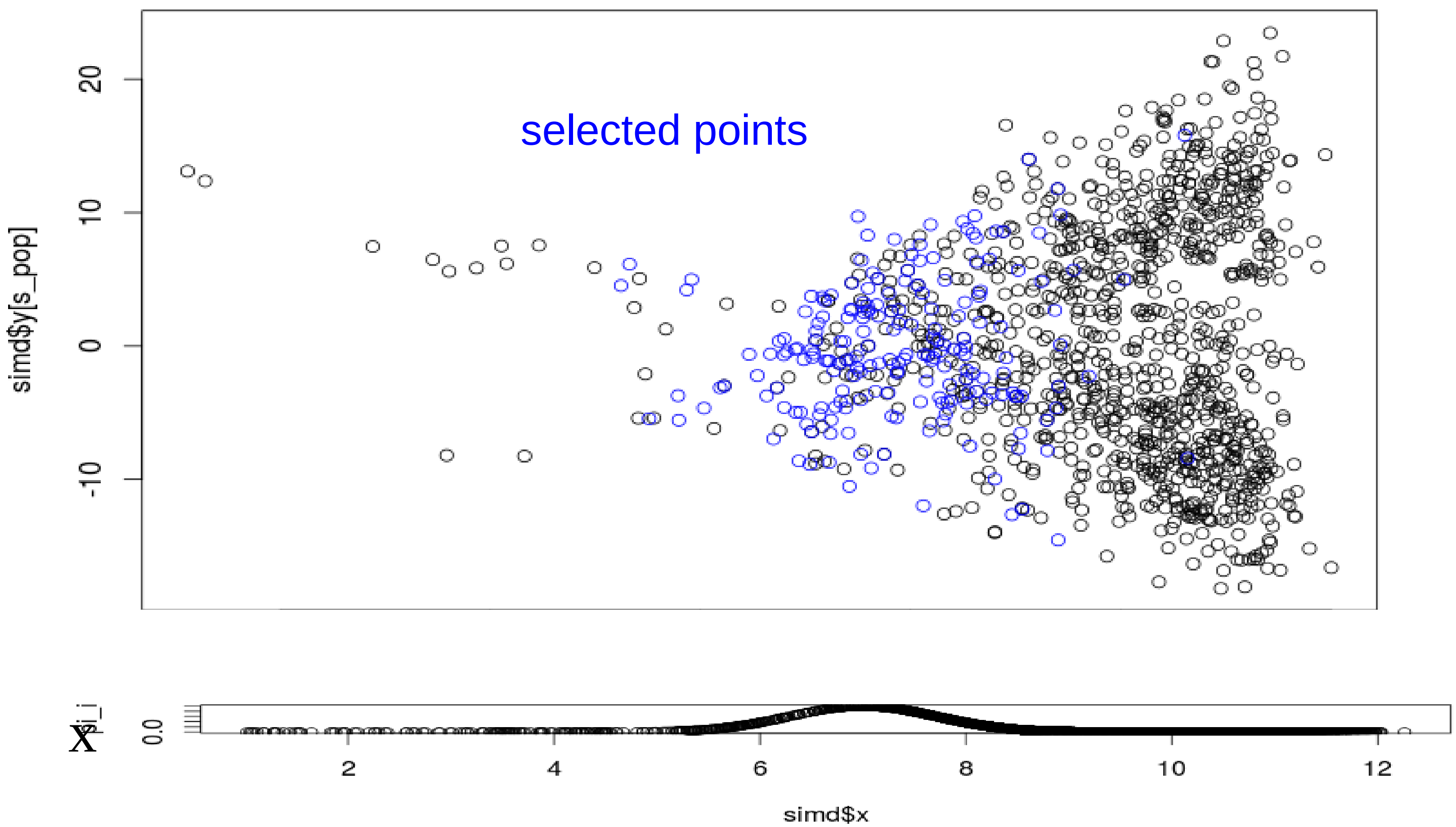


Unequal Probability of Selection



Draw sample with probability of selection proportional to

Unequal Probability of Selection



Trees Under Unequal Probability of Selection

Ignore

rpms(y~va+vb+vc+ve+vf, data=simd, e_eq=y~x)

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

Estimating Equation
y ~ x

   Splits
1
va %in% c('2','3')
va %in% c('2','3') & vc %in% c('4','3','1','0')


    coefficients
node          1          x
   4  16.100792 -2.218358
   5   9.708693 -1.587803
   3 -17.017023  2.384377
```

Weight

>rpms(y~va+vb+vc+vd+ve+vf, data=simd, e_eq=y~x, weights=1/pi_i)

```
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  26.94646 -2.99273
   3 -21.78096  2.46555
```

Trees Under Unequal Probability of Selection

Ignore

rpms(y~va+vb+vc+ve+vf, data=simd, e_eq=y~x)

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

Estimating Equation
y ~ x

   Splits
1
va %in% c('2','3')
va %in% c('2','3') & vc %in% c('4','3','1','0')


    coefficients
node          1         x
   4  16.100792 -2.218358
   5   9.708693 -1.587803
   3 -17.017023  2.384377
```

includes vc

Weight

>rpms(y~va+vb+vc+vd+ve+vf, data=simd, e_eq=y~x, weights=1/pi_i)

```
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  26.94646 -2.99273
   3 -21.78096  2.46555
```

Trees Under Unequal Probability of Selection

Ignore

rpms(y~va+vb+vc+ve+vf, data=simd, e_eq=y~x)

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

Estimating Equation
y ~ x

   Splits
1
va %in% c('2','3')
va %in% c('2','3') & vc %in% c('4','3','1','0')


    coefficients
node          1         x
   4  16.100792 -2.218358
   5   9.708693 -1.587803
   3 -17.017023  2.384377
```

Weight

>rpms(y~va+vb+vc+vd+ve+vf, data=simd, e_eq=y~x, weights=1/pi_i)

```
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  26.94646 -2.99273
   3 -21.78096  2.46555
```

correct model

Predict Method

predict.rpms {rpms}

predict.rpms

Description

Predicted values based on rpms object

Usage

```
## S3 method for class 'rpms'
predict(object, newdata, ...)
```

Arguments

`object`	Object inheriting from rpms
`newdata`	data frame with variables to use for predicting new values.
`...`	further arguments passed to or from other methods.

Value

vector of predicticed values for each row of newdata

Examples

```
{
# get rpms model of mean retirement contribution by several factors
r1 <- rpms(FINDRETX~EDUC_REF+AGE_REF+BLS_URBN+REGION, data = CE)

# first 10 predicted means
predict(r1, CE[1:10, ])

}
```

Predict Method

```
> predict(rx1, newdata)
```

Predict Method

```
> predict(rx1, newdata)
```

object of type rpms

Predict Method

```
> predict(rx1, newdata)
```

dataframe containing all variables on the right hand side of rp_equ and e_equ

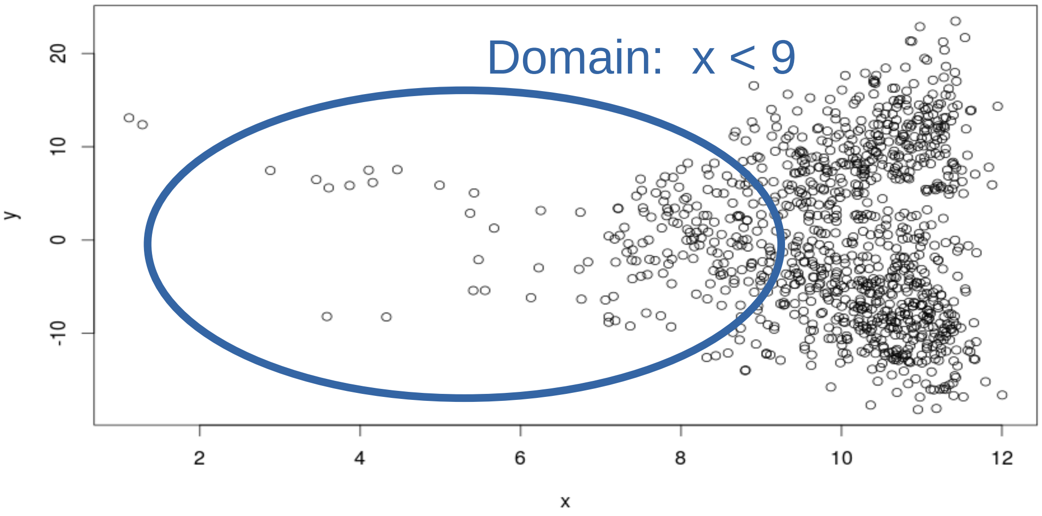
Predict Method

> predict(rx1, newdata)

```
> pre1 <- sample(10000, 8)
> new <- as.data.frame.array(simd[pre1, -c(1, 2, 10)],
row.names=1:8)
> new[,"x"] <- round(new[,"x"], 2)
>
> predict(iid, newdata=new)
[1]  3.191847  3.191847  3.191847  3.191847  3.191847 -1.759392
3.191847  3.191847
```

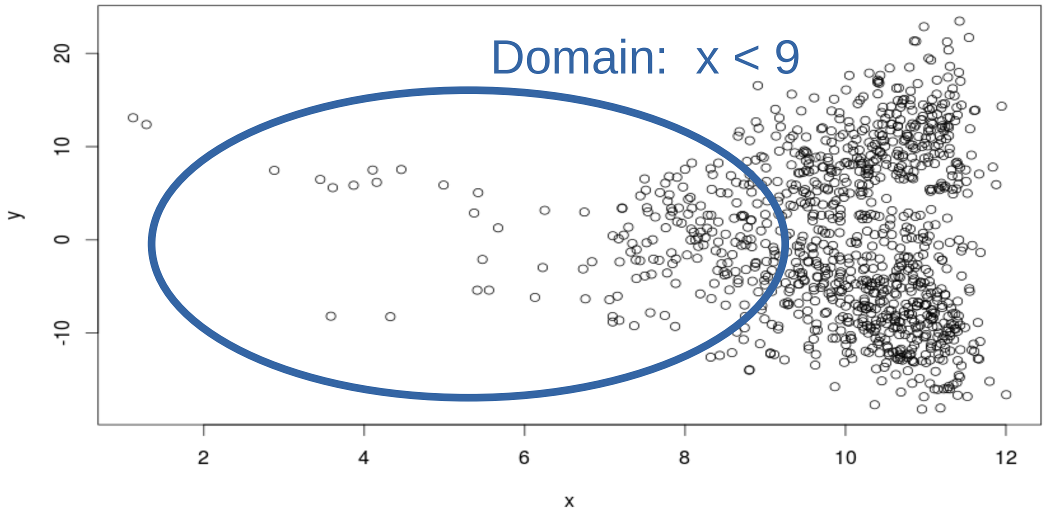
Domain Estimates

rpms models can be used for domain estimates using predict



Domain Estimates

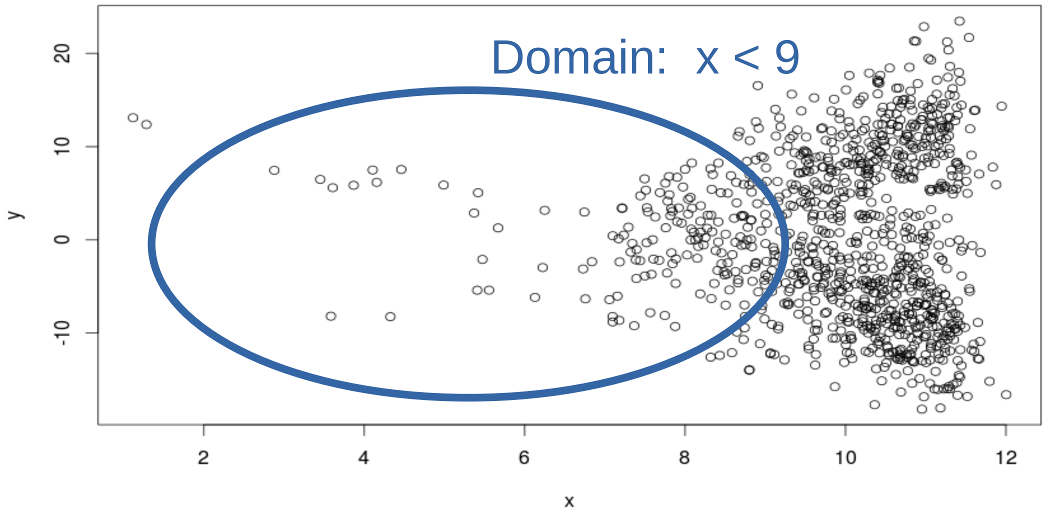
rpms models can be used for domain estimates using predict




```
D <- which(simd[,"x"]<9)

mean(predict(rxp, simd[D,])) + est_res
```

Domain Estimates

rpms models can be used for domain estimates using predict

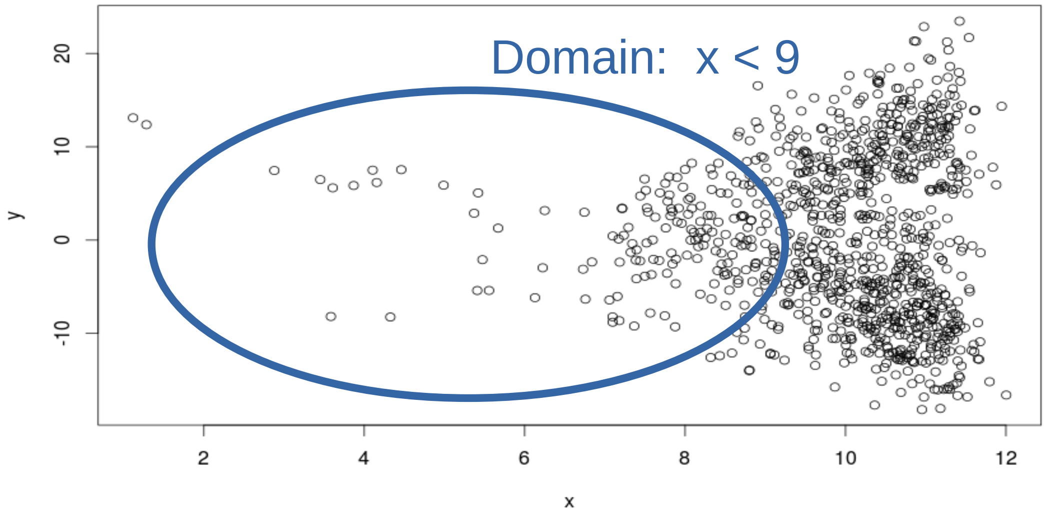


predicted population values

```
D <- which(simd[,"x"]<9)

mean(predict(rxp, simd[D,])) + est_res
```

Domain Estimates

rpms models can be used for domain estimates using predict

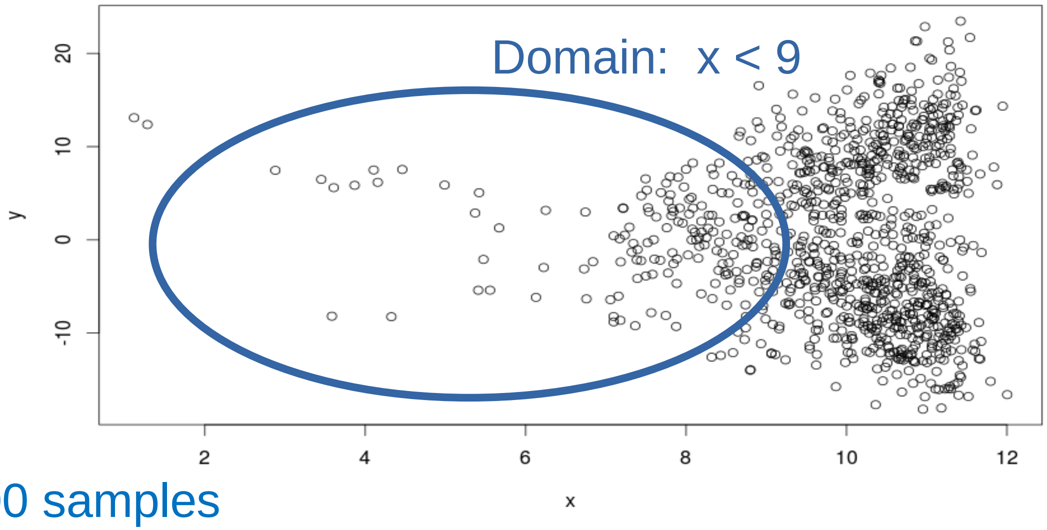



```
D <- which(simd[,"x"]<9)

mean(predict(rxp, simd[D,])) + est_res
```

HT estimate of population residuals

Domain Estimates

rpms models can be used for domain estimates using predict



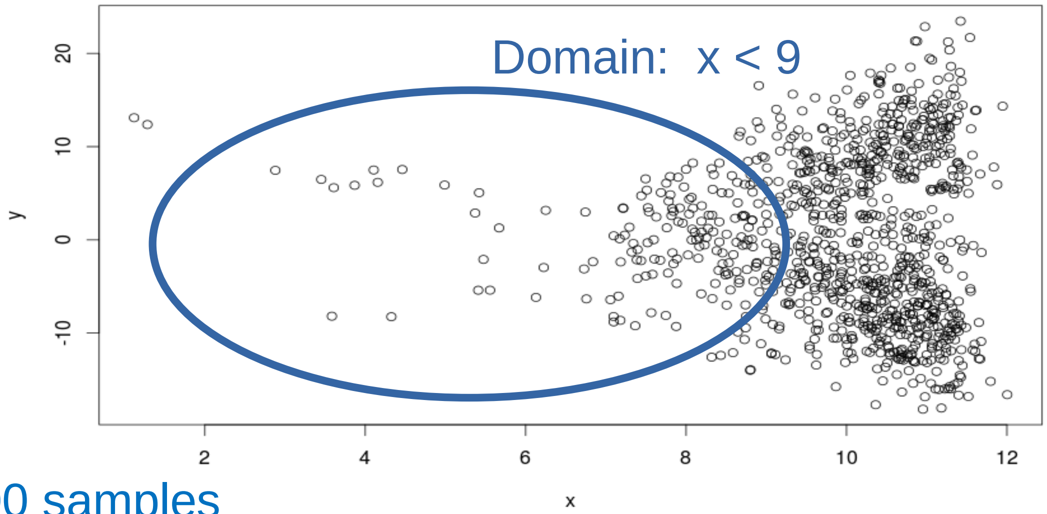
over 500 samples

Mean of HT estimate: -0.7300458

Mean using predict : -0.7710101

Domain Estimates

rpms models can be used for domain estimates using predict



over 500 samples

Mean of HT estimate: -0.7300458

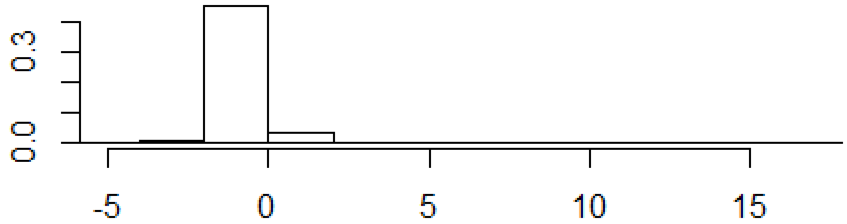
Mean using predict : -0.7710101

Population mean: -0.7622469

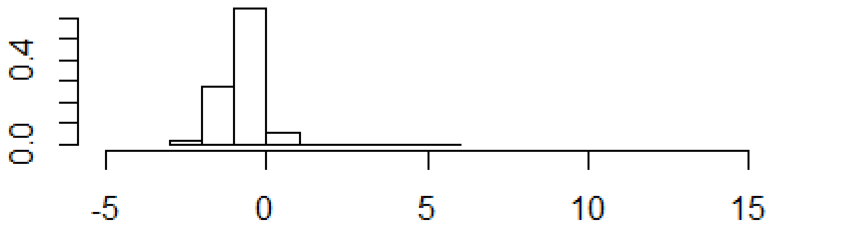
Domain Estimates

over 500 samples

usual HT estimator
sd = 1.019409



estimator using predict
sd = 0.5434727



Other rpms Options

rpms

Description

main function producing a regression tree using variables from rp_equ to partition the data and fit the model e_equ on each node. Currently only uses data with complete cases.

Usage

```
rpms(rp_equ, data, weights = ~1, strata = ~1, clusters = ~1, e_equ = ~1,
  e_fn = "survLm", l_fn = NULL, bin_size = NULL, perm_reps = 500L,
  pval = 0.05)
```

Arguments

rp_equ	formula containing all variables for partitioning
data	data.frame that includes variables used in rp_equ, e_equ, and design information
weights	formula or vector of sample weights for each observation
strata	formula or vector of strata labels
clusters	formula or vector of cluster labels
e_equ	formula for modeling data in each node
e_fn	string name of function to use for modeling (only "survLm" is operational)
l_fn	loss function (does nothing yet)
bin_size	numeric minimum number of observations in each node
perm_reps	integer specifying the number of permuations
pval	numeric p-value used to reject null hypothesis in permutation test

Value

object of class "rpms"

Examples

Other rpms Options

rpms has a number of other optional arguments:

l_fn	loss-function written in R
bin_size	minimum number of observations in each node
pval	p-value used to reject null hypothesis

CE Data

CE {rpms} R Documentation

CE Consumer expenditure data (first quarter of 2014)

Description

A dataset containing consumer unit characteristics, assets and expenditure data from the Bureau of Labor Statistics' Consumer Expenditure Survey public use interview data file.

Usage

CE

Format

A data frame with 6483 rows and 61 variables:

Location and sample-design variables

NEWID

Consumer unit identifying variable

PSU

CE Data

data(CE)

FSALARYX: Income from Salary
FINCBTAX: Income Before Tax
FINDRETX: Amount put in an individual retirement plan
FAM_SIZE: Number of members
NO_EARNR: Number of earners
PERSOT64: Number of people >64 yrs old
CUTENURE: 1 Owned with mortgage; 2-6 Other
VEHQ: Number of owned vehicles
REGION: 1 Northeast; 2 Midwest; 3 South; 4 West

CE Example

```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000), ]
```

CE Example

```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000),]
```

Households with income from salary

and

Income < $600,000

CE Example

```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000), ]
```

subset of
CE dataset

Households with income from salary

and

Income < $600,000

CE Example

```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000), ]
```

```
workers$saver = ifelse(workers$FINDRETX>0, 1, 0)
```

new variable

1 if has retirement savings

0 otherwise

CE Example

```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000), ]

workers$saver = ifelse(workers$FINDRETX>0, 1, 0)

rpms(rp_equ=saver~FAM_SIZE+NO_EARNR+CUTENURE+VEHQ+REGION,
     e_equ = saver~FINCBTAX,
     weights=~FINLWT21, clusters=~CID, data=workers, pval=.01)
```

CE Example

```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000), ]

workers$saver = ifelse(workers$FINDRETX>0, 1, 0)
```

model savers as function of family income

```
rpms(rp_equ=saver~FAM_SIZE+NO_EARNR+CUTENURE+VEHQ+REGION,
     e_equ = saver~FINCBTAX,
     weights=~FINLWT21, clusters=~CID, data=workers, pval=.01)
```

CE Example

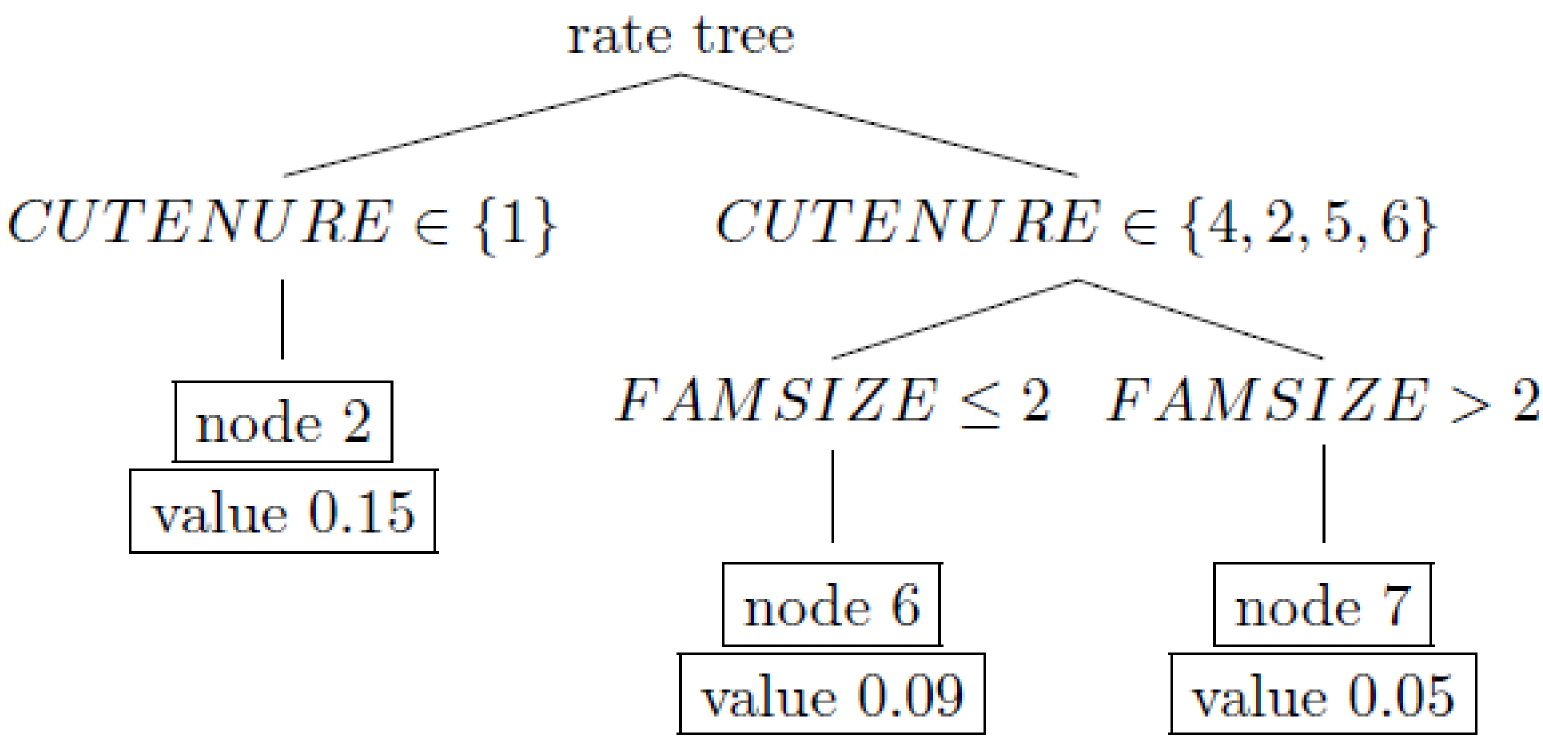
```
workers = CE[which(CE$FSALARYX>0 & CE$FINCBTAX<600000), ]

workers$saver = ifelse(workers$FINDRETX>0, 1, 0)
```

model savers as function of family income

design information

```
rpms(rp_equ=saver~FAM_SIZE+NO_EARNR+CUTENURE+VEHQ+REGION,
     e_equ = saver~FINCBTAX,
     weights=~FINLWT21, clusters=~CID, data=workers, pval=.01)
```

CE Example


rate tree
$CUTENURE \in \{1\}$
$CUTENURE \in \{4, 2, 5, 6\}$
node 2
value 0.15
$FAMSIZE \leq 2$
$FAMSIZE > 2$
node 6
value 0.09
node 7
value 0.05

CE Example

RPMS Recursive Partitioning Equation

saver ~ FAM_SIZE + NO_EARNR + PERSOT64 + CUTENURE + VEHQ + REGION

Estimating Equation

saver ~ FINCBTAX

```
     Splits
[1,]  1
[2,]   CUTENURE %in% c('1')
[3,]   CUTENURE %in%  c('4','2','5','6') & FAM_SIZE <= 2

    coefficients
node            1            FINCBTAX
   2  0.068209137      7.805248e-07
   6  0.011400909      1.548897e-06
   7 -0.004685768      8.678787e-07
```

CE Example

RPMS Recursive Partitioning Equation

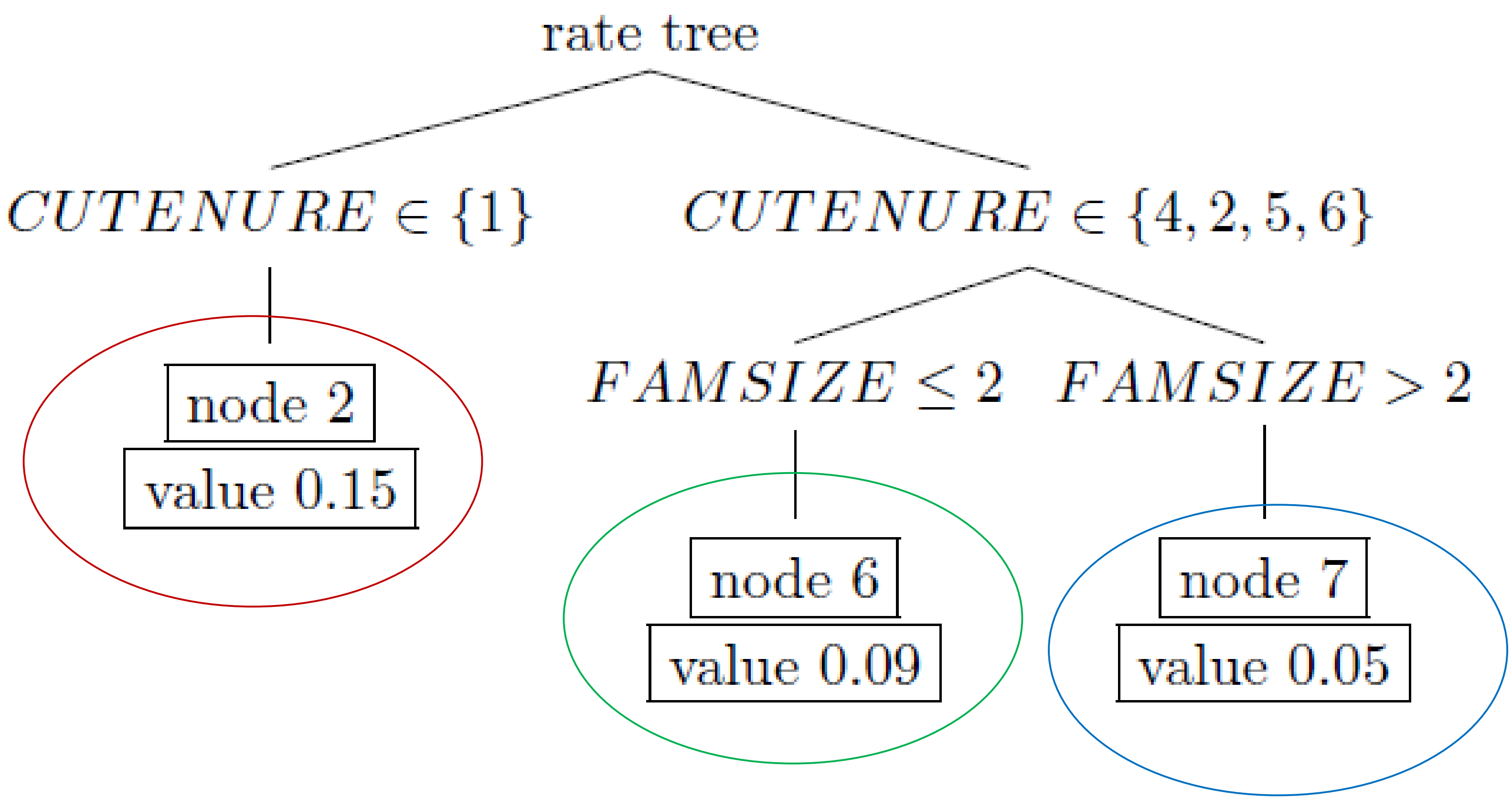
saver ~ FAM_SIZE + NO_EARNR + PERSOT64 + CUTENURE + VEHQ + REGION

Estimating Equation

saver ~ FINCBTAX

```
     Splits
[1,]  1
[2,]   CUTENURE %in% c('1')
[3,]   CUTENURE %in%  c('4','2','5','6') & FAM_SIZE <= 2

    coefficients
node            1            FINCBTAX
   2  0.068209137      7.805248e-07
   6  0.011400909      1.548897e-06
   7 -0.004685768      8.678787e-07
```

CE Example


rate tree
CUTENURE ∈ {1}
CUTENURE ∈ {4, 2, 5, 6}
node 2
value 0.15
FAMSIZE ≤ 2
FAMSIZE > 2
node 6
value 0.09
node 7
value 0.05

CE Example

Use in_node function

high = in_node(6, rate_tree, data=workers)

CE Example

Use in_node function

high = in_node(6, rate_tree, data=workers)

node

rpms
object

could be
new data

index of observations
in a given node

CE Example

Use in_node function

high = in_node(6, rate_tree, data=workers)

index of observations
in a given node

node

rpms
object

could be
new data

Can be used to analyze groups separately

Example: summary(workers[high,])

CE Example

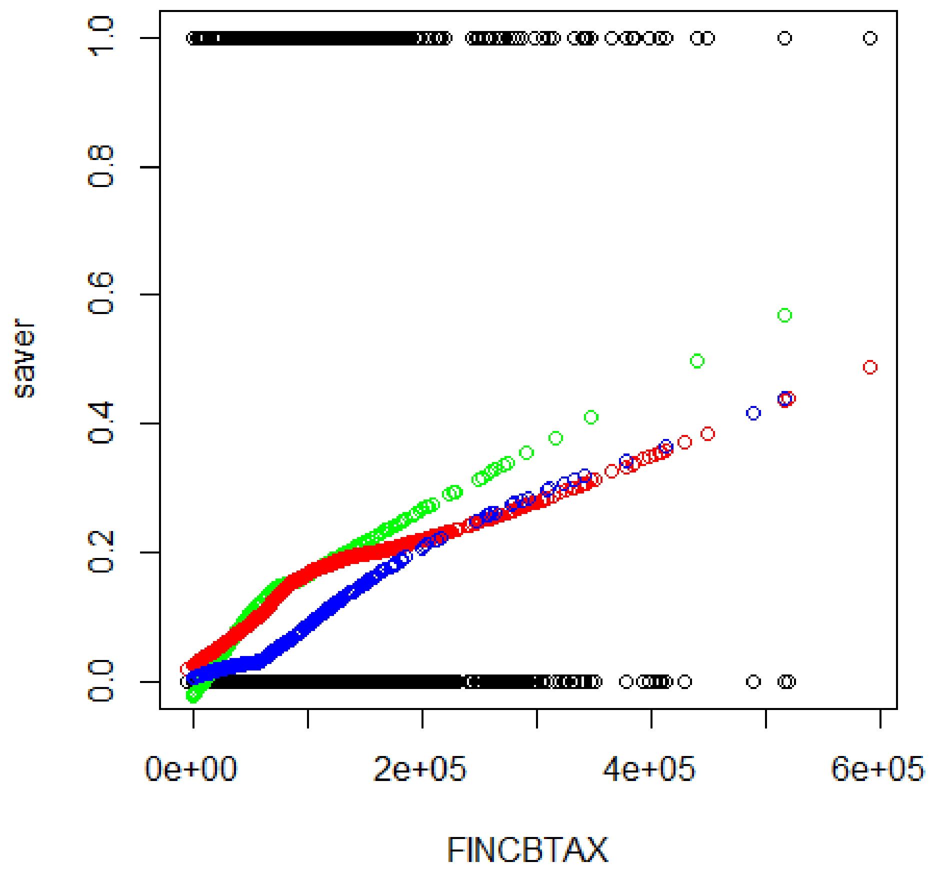
Use in_node function

high = in_node(6, rate_tree, data=workers)

mort = in_node(2, rate_tree, data=workers)

kids = in_node(7, rate_tree, data=workers)

CE Example



Wrap Up

- We have introduced the rpms package and demonstrated a number of functions including

Wrap Up

- We have introduced the rpms package and demonstrated a number of functions including

- The package does NOT yet:
  - have a general plot function
  - handle missing values

Wrap Up

- We have introduced the rpms package and demonstrated a number of functions including

- The package does NOT yet:
  - have a general plot function
  - handle missing values

- This is a working package: version: 0_2_0

Wrap Up

- We have introduced the rpms package and demonstrated a number of functions including

- The package does NOT yet:
  - have a general plot function
  - handle missing values

Note the 0

- This is a working package: version: 0_2_0
  - Lots of features are experimental and being tested (don’t use)
  - May have bugs (weights not used in variable selection in 0_2_0)
  - Working on more features and applications
  - Open to taking suggestions

Wrap Up

- We have introduced the rpms package and demonstrated a number of functions including

- The package does NOT yet:
  - have a general plot function
  - handle missing values

Note the 0

- This is a working package: version: 0_2_0
  - Lots of features are experimental and being tested (don't use)
  - May have bugs (weights not used in variable selection in 0_2_0)
  - Working on more features and applications
  - Open to taking suggestions

  - Look for version 0_3_0 in early April

Thank You

toth.daniell@bls.gov