Decision Trees

INSOFE Lab Activity on Decision Trees

23 July 2017

Contents

C5.0 Trees **1**
- Goal . . . 1
- Agenda . . . 2
- Reading & Understanding the Data . . . 2
  - Read the Data . . . 2
  - Understand the data . . . 2
- Data Pre-processing . . . 4
  - Verify Data Integrity . . . 4
  - Split the Data into train and test sets . . . 4
  - Impute the missing values . . . 5
- Build a Decision Tree . . . 5
  - Model the tree . . . 5
  - Variable Importance in trees . . . 5
  - Rules from trees . . . 6
  - Plotting the tree . . . 7
- Evaluating the model . . . 8
  - Predictions on the test data . . . 8

CART Trees **9**
- Goal . . . 9
- Agenda . . . 9
- Reading & Understanding the Data . . . 9
  - Read the Data . . . 9
  - Understand the data . . . 9
- Data Pre-processing . . . 11
  - Verify Data Integrity . . . 11
  - Split the Data . . . 11
- Build a Regression Tree . . . 12
  - Model the tree . . . 12
  - Tree Explicability . . . 12
- Evaluation on Test Data . . . 13

C5.0 Trees

NOTE Before starting this assignment please remember to clear your environment, you can do that by running the following code chunk

```
rm(list=ls(all=TRUE))
```

Goal

- The goal of this activity is to predict wether a patient has liver disease or not based on various patient related attributes

Agenda

- Get the data
- Data Pre-processing
- Build a model
- Predictions
- Communication

Reading & Understanding the Data

Read the Data

Make sure the dataset is located in your current working directory, or else you can change your working directory using the "setwd()" function.

```
setwd("F:/INSOFE/MachineLearning/Week8/LabDecisionTree")
ilpd_data <- read.csv("ilpd_data.csv")
```

Understand the data

- Use the str(), summary(), head() and tail() functions to get the dimensions and types of attributes in the dataset
- The dataset has 582 observations and 11 variables
- **The variable descriptions are given below:**

1 - age : Age of the patient

2 - gender : Gender of the patient

3 - TB : Total Bilirubin content

4 - DB : Direct Bilirubin content

5 - alk_phos : Alkaline Phosphotase content

6 - alamine : Alamine Aminotransferase content

7 - aspartate : Aspartate Aminotransferase content

8 - TP : Total Protiens content

9 - albumin : Albumin content

10 - A/G : Ratio of Albumin and Globulin

11 - Disease : Whether the patient has liver disease or not

```
str(ilpd_data)
```

```
## 'data.frame':    582 obs. of  11 variables:
##  $ age      : int  62 62 58 72 46 26 29 17 55 57 ...
##  $ gender   : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 1 1 2 2 2 ...
##  $ TB       : num  10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 0.6 ...
##  $ DB       : num  5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 0.1 ...
##  $ alk_phos : int  699 490 182 195 208 154 202 202 290 210 ...
##  $ alamine  : int  64 60 14 27 19 16 14 22 53 51 ...
```

```
##  $ aspartate: int  100 68 20 59 14 12 11 19 58 59 ...
##  $ TP       : num  7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 5.9 ...
##  $ albumin  : num  3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 2.7 ...
##  $ A.G      : num  0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 0.8 ...
##  $ disease  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 1 2 2 ...
```

```
summary(ilpd_data)
```

```
##       age           gender          TB               DB
##  Min.   : 4.00   Female:141   Min.   : 0.400   Min.   : 0.100
##  1st Qu.:33.00   Male  :441   1st Qu.: 0.800   1st Qu.: 0.200
##  Median :45.00                Median : 1.000   Median : 0.300
##  Mean   :44.71                Mean   : 3.303   Mean   : 1.488
##  3rd Qu.:57.75                3rd Qu.: 2.600   3rd Qu.: 1.300
##  Max.   :90.00                Max.   :75.000   Max.   :19.700
##
##     alk_phos         alamine          aspartate            TP
##  Min.   :  63.0   Min.   :  10.00   Min.   :  10.0   Min.   :2.700
##  1st Qu.: 175.2   1st Qu.:  23.00   1st Qu.:  25.0   1st Qu.:5.800
##  Median : 208.0   Median :  35.00   Median :  42.0   Median :6.600
##  Mean   : 290.8   Mean   :  80.82   Mean   : 110.1   Mean   :6.483
##  3rd Qu.: 298.0   3rd Qu.:  60.75   3rd Qu.:  87.0   3rd Qu.:7.200
##  Max.   :2110.0   Max.   :2000.00   Max.   :4929.0   Max.   :9.600
##
##     albumin           A.G         disease
##  Min.   :0.900   Min.   :0.3000   no :167
##  1st Qu.:2.600   1st Qu.:0.7000   yes:415
##  Median :3.100   Median :0.9400
##  Mean   :3.142   Mean   :0.9471
##  3rd Qu.:3.800   3rd Qu.:1.1000
##  Max.   :5.500   Max.   :2.8000
##                  NA's   :4
```

```
head(ilpd_data)
```

```
##   age gender   TB  DB alk_phos alamine aspartate  TP albumin  A.G disease
## 1  62   Male 10.9 5.5      699      64       100 7.5     3.2 0.74     yes
## 2  62   Male  7.3 4.1      490      60        68 7.0     3.3 0.89     yes
## 3  58   Male  1.0 0.4      182      14        20 6.8     3.4 1.00     yes
## 4  72   Male  3.9 2.0      195      27        59 7.3     2.4 0.40     yes
## 5  46   Male  1.8 0.7      208      19        14 7.6     4.4 1.30     yes
## 6  26 Female  0.9 0.2      154      16        12 7.0     3.5 1.00     yes
```

```
tail(ilpd_data)
```

```
##     age gender   TB  DB alk_phos alamine aspartate  TP albumin  A.G
## 577  32   Male 12.7 8.4      190      28        47 5.4     2.6 0.90
## 578  60   Male  0.5 0.1      500      20        34 5.9     1.6 0.37
## 579  40   Male  0.6 0.1       98      35        31 6.0     3.2 1.10
## 580  52   Male  0.8 0.2      245      48        49 6.4     3.2 1.00
## 581  31   Male  1.3 0.5      184      29        32 6.8     3.4 1.00
## 582  38   Male  1.0 0.3      216      21        24 7.3     4.4 1.50
##     disease
## 577     yes
## 578      no
## 579     yes
```

```
## 580     yes
## 581     yes
## 582      no
```

Data Pre-processing

Verify Data Integrity

- Verify if the dataset has missing values

```
sum(is.na(ilpd_data))
```

```
## [1] 4
```

- Verify the data types assigned to the variables in the dataset

```
str(ilpd_data)
```

```
## 'data.frame':    582 obs. of  11 variables:
##  $ age      : int  62 62 58 72 46 26 29 17 55 57 ...
##  $ gender   : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 1 1 2 2 2 ...
##  $ TB       : num  10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 0.6 ...
##  $ DB       : num  5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 0.1 ...
##  $ alk_phos : int  699 490 182 195 208 154 202 202 290 210 ...
##  $ alamine  : int  64 60 14 27 19 16 14 22 53 51 ...
##  $ aspartate: int  100 68 20 59 14 12 11 19 58 59 ...
##  $ TP       : num  7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 5.9 ...
##  $ albumin  : num  3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 2.7 ...
##  $ A.G      : num  0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 0.8 ...
##  $ disease  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 1 2 2 ...
```

```
# Dependent variable is 'disease' and independent variables are discrete and categorical variables
```

Split the Data into train and test sets

- Use stratified sampling to split the data into train/test sets (70/30)
- Use the createDataPartition() function from the caret package to do stratified sampling

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
# Set the seed after attaching the caret package

set.seed(007)

# The first argument is the imbalanced class reference variable, the second is the proportion to sample

# Remember to include list = F as the function returns a list otherwise which would not be able to subs

trainRows <- createDataPartition(ilpd_data$disease, p = .7, list = F)

train_df <- ilpd_data[trainRows, ]
```

```
test_df <- ilpd_data[-trainRows, ]
```

Impute the missing values

- Impute missing values using knnImputation() function in both the train and test datasets

```
library(DMwR)
```

```
## Loading required package: grid
```

```
train_df <- knnImputation(train_df)
test_df <- knnImputation(test_df)
#Missing values are imputed with knnImputation function
sum(is.na(train_df))
```

```
## [1] 0
```

```
sum(is.na(test_df))
```

```
## [1] 0
```

Build a Decision Tree

Model the tree

- Use Quinlan's C5.0 decision tree algorithm implementation from the C50 package to build your decision tree

```
library(C50)

c5_entropy <- C5.0(disease ~ . , train_df)
```

- Build a rules based tree

```
# Use the rules = T argument if you want to extract rules later from the model

c5_entropy_rules <- C5.0(disease ~ . , train_df, rules = T)
```

Variable Importance in trees

- Find the importance of each variable in the dataset

```
C5imp(c5_entropy, metric = "usage")
```

```
##           Overall
## DB         100.00
## alk_phos    71.81
## alamine     66.67
## TB          41.67
## gender      38.48
## TP          24.51
## albumin     11.27
## age          8.82
## aspartate    7.11
## A.G          1.96
```

Rules from trees

- Understand the summary of the returned c5.0 rules based on the decision tree model

```
summary(c5_entropy_rules)
```

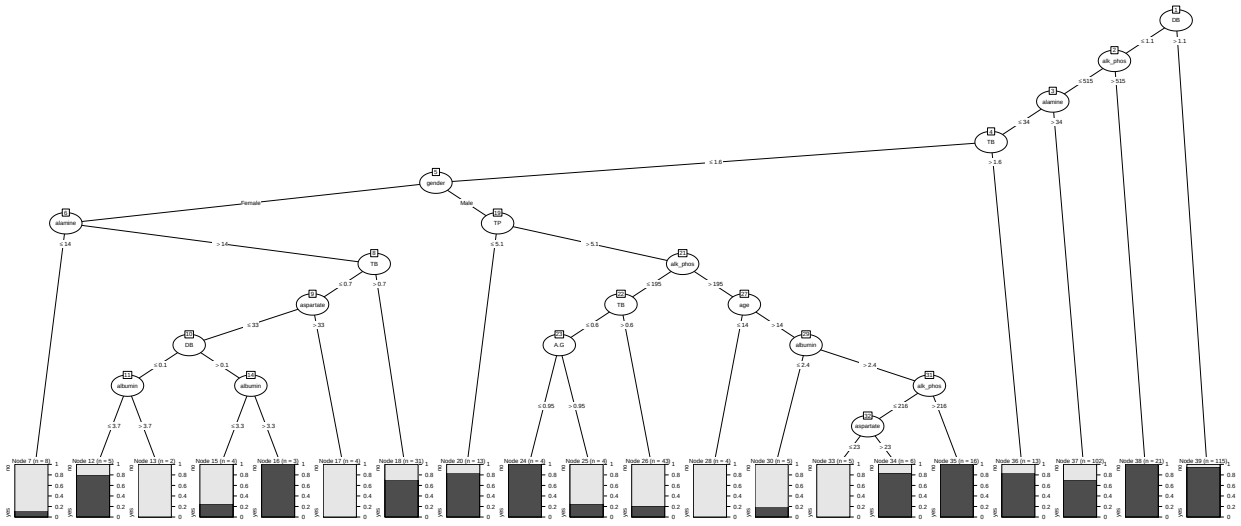
```
##
## Call:
## C5.0.formula(formula = disease ~ ., data = train_df, rules = T)
##
##
## C5.0 [Release 2.07 GPL Edition]      Sat Aug 05 20:33:34 2017
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 408 cases (11 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (5, lift 3.0)
##  gender = Male
##  TB <= 1.6
##  alk_phos > 195
##  alk_phos <= 216
##  aspartate <= 23
##  ->  class no  [0.857]
##
## Rule 2: (4, lift 2.9)
##  DB <= 0.1
##  aspartate <= 33
##  albumin > 3.7
##  ->  class no  [0.833]
##
## Rule 3: (4, lift 2.9)
##  gender = Female
##  TB <= 0.7
##  alamine <= 34
##  aspartate > 33
##  ->  class no  [0.833]
##
## Rule 4: (4, lift 2.9)
##  age <= 14
##  gender = Male
##  alk_phos <= 515
##  alamine <= 34
##  ->  class no  [0.833]
##
## Rule 5: (8/1, lift 2.8)
##  gender = Female
##  alamine <= 14
##  ->  class no  [0.800]
##
## Rule 6: (43/9, lift 2.7)
##  gender = Male
```

```
##  TB > 0.6
##  TB <= 1.6
##  alk_phos <= 195
##  alamine <= 34
##  TP > 5.1
##  ->  class no  [0.778]
##
## Rule 7: (152/13, lift 1.3)
##  TB > 1.6
##  ->  class yes  [0.909]
##
## Rule 8: (389/103, lift 1.0)
##  alamine > 14
##  ->  class yes  [0.734]
##
## Default class: yes
##
##
## Evaluation on training data (408 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##       8   69(16.9%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##      58    59    (a): class no
##      10   281    (b): class yes
##
##
##  Attribute usage:
##
##   99.02% alamine
##   50.00% TB
##   15.69% gender
##   12.75% alk_phos
##   10.54% TP
##    3.19% aspartate
##    0.98% age
##    0.98% DB
##    0.98% albumin
##
##
## Time: 0.0 secs
```

Plotting the tree

- Call the plot function on the tree object to visualize the tree

```
plot(c5_entropy)
```



Evaluating the model

Predictions on the test data

- Evaluate the decision tree using the standard error metrics on test data

```
predicted <- predict(c5_entropy, test_df)
```

- Report error metrics for classification on test data

```
library(caret)
library(e1071)

confusionMatrix(predicted, test_df$disease)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no   19  21
##        yes  31 103
##
##                Accuracy : 0.7011
##                  95% CI : (0.6272, 0.7681)
##     No Information Rate : 0.7126
##     P-Value [Acc > NIR] : 0.6658
##
##                   Kappa : 0.224
##  Mcnemar's Test P-Value : 0.2120
##
##             Sensitivity : 0.3800
##             Specificity : 0.8306
##          Pos Pred Value : 0.4750
##          Neg Pred Value : 0.7687
##              Prevalence : 0.2874
##          Detection Rate : 0.1092
##    Detection Prevalence : 0.2299
```

```
##       Balanced Accuracy : 0.6053
##
##        'Positive' Class : no
##
```

CART Trees

NOTE Before starting this assignment please remember to clear your environment, you can do that by running the following code chunk

```
rm(list=ls(all=TRUE))
```

- The classification and regression trees use gini index in place of the gain ratio (based on information gain) used by the ID3 based algorithms, such as c4.5 and c5.0

Goal

- The goal of this activity is to predict the heating load of a residential building, if the building parameters are given
- Hence, in the future architects would be able to build more energy efficient buildings as they can optimize the building parameters to reduce the heating load

Agenda

- Get the data
- Data Pre-processing
- Build a model
- Predictions
- Communication

Reading & Understanding the Data

Read the Data

- Make sure the dataset is located in your current working directory, or else you can change your working directory using the "setwd()" function.

```
setwd("F:/INSOFE/MachineLearning/Week8/LabDecisionTree")
energy_data <- read.csv("building_energy.csv", na.strings = "")
```

Understand the data

- Use the str(), summary(), head() and tail() functions to get the dimensions and types of attributes in the dataset
- The dataset has 768 observations and 9 variables

```
str(energy_data)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ relative_compactness     : num  0.98 0.98 0.98 0.98 0.9 0.9 0.9 0.9 0.86 0.86 ...
##  $ surface_area             : num  514 514 514 514 564 ...
##  $ wall_area                : num  294 294 294 294 318 ...
##  $ roof_area                : num  110 110 110 110 122 ...
##  $ overall_height           : num  7 7 7 7 7 7 7 7 7 7 ...
##  $ orientation              : int  2 3 4 5 2 3 4 5 2 3 ...
##  $ glazing_area             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ glazing_area_distribution: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ heating_load             : num  15.6 15.6 15.6 15.6 20.8 ...
```

```
summary(energy_data)
```

```
##  relative_compactness  surface_area     wall_area       roof_area
##  Min.   :0.6200       Min.   :514.5   Min.   :245.0   Min.   :110.2
##  1st Qu.:0.6825       1st Qu.:606.4   1st Qu.:294.0   1st Qu.:140.9
##  Median :0.7500       Median :673.8   Median :318.5   Median :183.8
##  Mean   :0.7642       Mean   :671.7   Mean   :318.5   Mean   :176.6
##  3rd Qu.:0.8300       3rd Qu.:741.1   3rd Qu.:343.0   3rd Qu.:220.5
##  Max.   :0.9800       Max.   :808.5   Max.   :416.5   Max.   :220.5
##  overall_height  orientation    glazing_area    glazing_area_distribution
##  Min.   :3.50   Min.   :2.00   Min.   :0.0000   Min.   :0.000
##  1st Qu.:3.50   1st Qu.:2.75   1st Qu.:0.1000   1st Qu.:1.750
##  Median :5.25   Median :3.50   Median :0.2500   Median :3.000
##  Mean   :5.25   Mean   :3.50   Mean   :0.2344   Mean   :2.812
##  3rd Qu.:7.00   3rd Qu.:4.25   3rd Qu.:0.4000   3rd Qu.:4.000
##  Max.   :7.00   Max.   :5.00   Max.   :0.4000   Max.   :5.000
##   heating_load
##  Min.   : 6.01
##  1st Qu.:12.99
##  Median :18.95
##  Mean   :22.31
##  3rd Qu.:31.67
##  Max.   :43.10
```

```
head(energy_data)
```

```
##   relative_compactness surface_area wall_area roof_area overall_height
## 1                 0.98        514.5     294.0    110.25              7
## 2                 0.98        514.5     294.0    110.25              7
## 3                 0.98        514.5     294.0    110.25              7
## 4                 0.98        514.5     294.0    110.25              7
## 5                 0.90        563.5     318.5    122.50              7
## 6                 0.90        563.5     318.5    122.50              7
##   orientation glazing_area glazing_area_distribution heating_load
## 1           2            0                         0       15.55
## 2           3            0                         0       15.55
## 3           4            0                         0       15.55
## 4           5            0                         0       15.55
## 5           2            0                         0       20.84
## 6           3            0                         0       21.46
```

```
tail(energy_data)
```

```
##     relative_compactness surface_area wall_area roof_area overall_height
## 763                 0.64        784.0     343.0     220.5            3.5
```

```
## 764                0.64        784.0     343.0     220.5            3.5
## 765                0.62        808.5     367.5     220.5            3.5
## 766                0.62        808.5     367.5     220.5            3.5
## 767                0.62        808.5     367.5     220.5            3.5
## 768                0.62        808.5     367.5     220.5            3.5
##     orientation glazing_area glazing_area_distribution heating_load
## 763           4          0.4                          5        18.16
## 764           5          0.4                          5        17.88
## 765           2          0.4                          5        16.54
## 766           3          0.4                          5        16.44
## 767           4          0.4                          5        16.48
## 768           5          0.4                          5        16.64
```

- The variable names are self explanatory, for further information visit http://www.sciencedirect.com/science/article/pii/S037877881200151X

Data Pre-processing

Verify Data Integrity

- Verify if the dataset has missing values

```
sum(is.na(energy_data))
```

```
## [1] 0
```

- Verify the data types assigned to the variables in the dataset

```
# Enter answer here

str(energy_data)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ relative_compactness     : num  0.98 0.98 0.98 0.98 0.9 0.9 0.9 0.9 0.86 0.86 ...
##  $ surface_area             : num  514 514 514 514 564 ...
##  $ wall_area                : num  294 294 294 294 318 ...
##  $ roof_area                : num  110 110 110 110 122 ...
##  $ overall_height           : num  7 7 7 7 7 7 7 7 7 7 ...
##  $ orientation              : int  2 3 4 5 2 3 4 5 2 3 ...
##  $ glazing_area             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ glazing_area_distribution: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ heating_load             : num  15.6 15.6 15.6 15.6 20.8 ...
```

Split the Data

- Split the data into train/test sets (70/30)

```
set.seed(123)

train_rows <- sample(1:nrow(energy_data), 0.7*nrow(energy_data))

train_cart <- energy_data[train_rows, ]

test_cart <- energy_data[-train_rows, ]
```

Build a Regression Tree

Model the tree

- Use the rpart package to build a cart tree to predict the heating load

```
library(rpart)

cart_gini <- rpart(heating_load ~ ., train_cart)

printcp(cart_gini)
```

```
##
## Regression tree:
## rpart(formula = heating_load ~ ., data = train_cart)
##
## Variables actually used in tree construction:
## [1] glazing_area         overall_height       relative_compactness
##
## Root node error: 54235/537 = 101
##
## n= 537
##
##         CP nsplit rel error   xerror      xstd
## 1 0.792696      0  1.000000 1.007002 0.0381495
## 2 0.083385      1  0.207304 0.208413 0.0146844
## 3 0.028539      2  0.123919 0.124980 0.0096109
## 4 0.013993      3  0.095380 0.096733 0.0071921
## 5 0.013968      4  0.081387 0.090938 0.0069360
## 6 0.010000      5  0.067418 0.073302 0.0059459
```

Tree Explicability

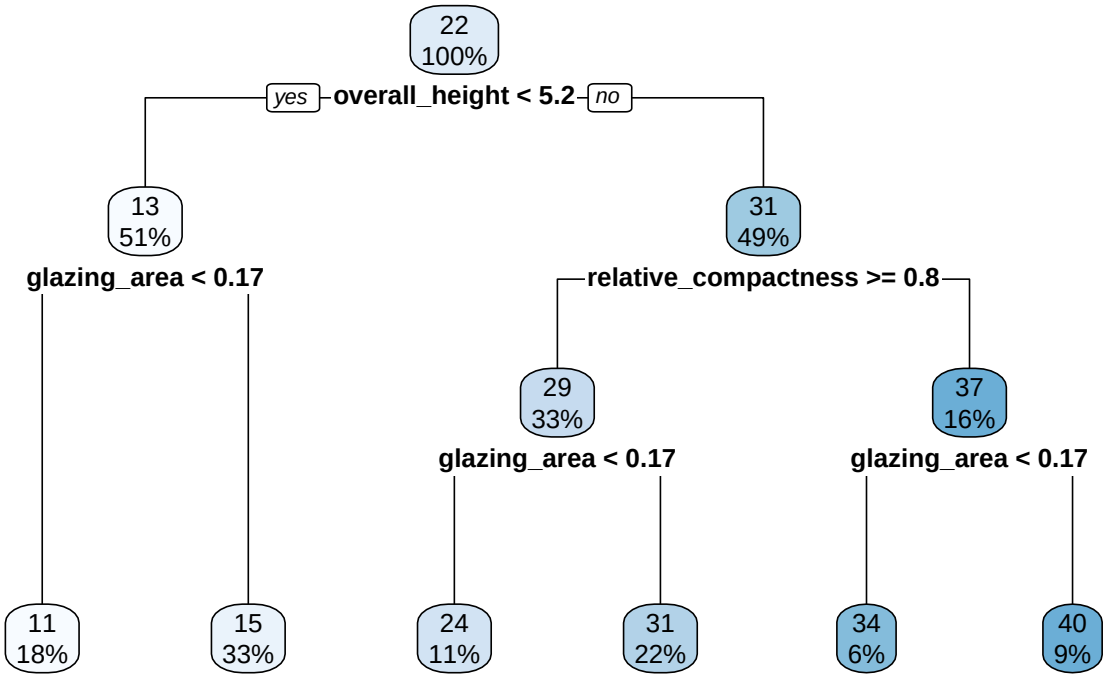
- Print the variable importance

```
cart_gini$variable.importance
```

```
##      relative_compactness              surface_area
##                 47514.5995                47514.5995
##             overall_height                 roof_area
##                 42992.1627                42992.1627
##                  wall_area              glazing_area
##                 17061.8176                   3064.3188
## glazing_area_distribution
##                    637.6964
```

- Plot the regression tree

```
library(rpart.plot)
library(RColorBrewer)
#fancyRpartPlot(cart_gini)
rpart.plot(cart_gini)
```



Evaluation on Test Data

- Report error metrics on the test data

```
predicted_cart <- predict(cart_gini, test_cart)
```

```
library(DMwR)

regr.eval(test_cart$heating_load, predicted_cart)
```

```
##       mae       mse      rmse      mape
## 2.0686987 7.2917648 2.7003268 0.1085194
```