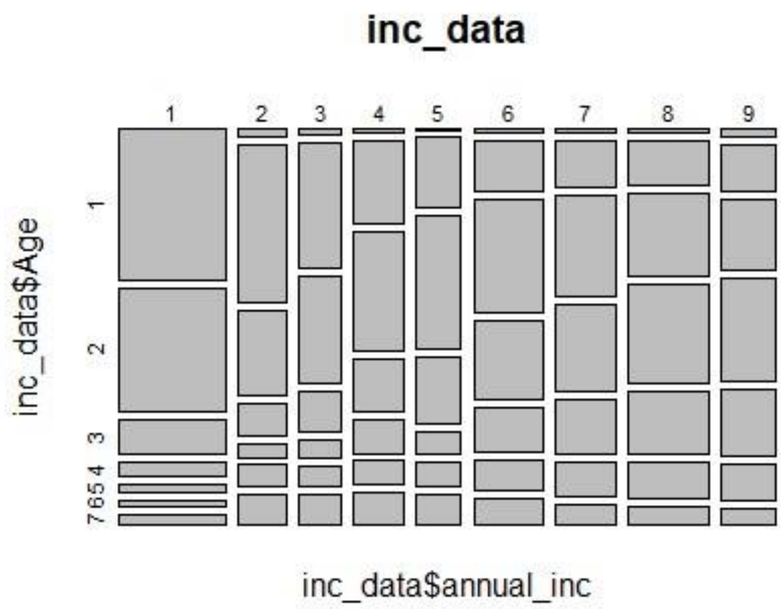
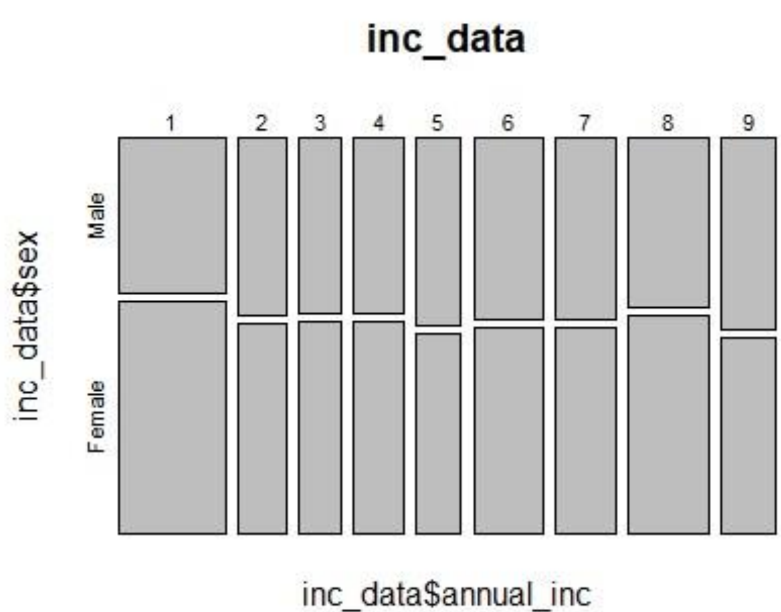
```
library(readxl)
inc_data <- read_excel("C:/Users/rakesh/Desktop/Business  data mining/Assignm
ent 2/income.data.xlsx",
                                col_names = FALSE, na = "NA") str(inc_data)

## Classes 'tbl_df', 'tbl' and 'data.frame':    8993 obs. of  14 variables:
##  $ X__1 : num  9 9 9 1 1 8 1 6 2 4 ...
##  $ X__2 : num  2 1 2 2 2 1 1 1 1 1 ...
##  $ X__3 : num  1 1 1 5 5 1 5 3 1 1 ...
##  $ X__4 : num  5 5 3 1 1 6 2 3 6 7 ...
##  $ X__5 : num  4 5 5 2 2 4 3 4 3 4 ...
##  $ X__6 : num  5 5 1 6 6 8 9 3 8 8 ...
##  $ X__7 : num  5 5 5 5 3 5 4 5 5 4 ...
##  $ X__8 : num  3 3 2 1 1 3 1 1 3 3 ...
##  $ X__9 : num  3 5 3 4 4 2 3 1 3 2 ...
##  $ X__10: num  0 2 1 2 2 0 1 0 0 0 ...
##  $ X__11: num  1 1 2 3 3 1 2 2 2 2 ...
##  $ X__12: num  1 1 3 1 1 1 3 3 3 3 ...
##  $ X__13: num  7 7 7 7 7 7 7 7 7 7 ... ##
$ X__14: num  NA 1 1 1 1 1 1 1 1 1 ...

colSums(is.na(inc_data))

##  X__1  X__2  X__3  X__4  X__5  X__6  X__7  X__8  X__9 X__10 X__11 X__12  ##
0     0   160     0    86   136   913     0   375     0   240   357
## X__13 X__14  ##
68   359

colnames(inc_data) <- c('annual_inc',
                      'sex', 'MaritalStatus',
                      'Age', 'Education', 'Occupation',
                      'duration','dual_inc',
                      'ppl_household','ppl_u18','HHStatus',
'home_type','Ethnicity','Language')
inc_data$annual_inc <- factor(inc_data$annual_inc,c("1","2","3","4","5","6","
7","8","9"))
str(inc_data$annual_inc)
```

```
##  Factor w/ 9 levels "1","2","3","4",..: 9 9 9 1 1 8 1 6 2 4 ...

inc_data$sex <- factor(inc_data$sex,c('1','2'),c('Male','Female'))
inc_data$MaritalStatus <- factor(inc_data$MaritalStatus)
inc_data$Age <- factor(inc_data$Age) inc_data$Education <-
factor(inc_data$Education) inc_data$Occupation <-
factor(inc_data$Occupation) inc_data$duration <-
factor(inc_data$duration) inc_data$dual_inc <-
factor(inc_data$dual_inc) inc_data$ppl_household <-
factor(inc_data$ppl_household) inc_data$ppl_u18 <-
factor(inc_data$ppl_u18) inc_data$HHStatus <-
factor(inc_data$HHStatus) inc_data$home_type <-
factor(inc_data$home_type) inc_data$Ethnicity <-
factor(inc_data$Ethnicity) inc_data$Language <-
factor(inc_data$Language) summary(inc_data)

##    annual_inc       sex       MaritalStatus Age      Education
##  1       :1745   Male  :4075   1    :3334     1: 878   1   : 264
##  8       :1308   Female:4918   2    : 668     2:2129   2   :1046
##  6       :1110                 3    : 875     3:2249   3   :2041
##  7       : 969                 4    : 302     4:1615   4   :3066
##  9       : 884                 5    :3654     5: 922   5   :1524
##  4       : 813                 NA's: 160     6: 640   6   : 966
##  (Other):2164                                7: 560   NA's:  86
##    Occupation   duration    dual_inc ppl_household     ppl_u18
##  1       :2820   1   : 270   1:5438   2       :2664   0       :5724
##  6       :1489   2   :1042   2:2211   3       :1670   1       :1506
##  4       :1062   3   : 686   3:1344   1       :1620   2       :1148
##  2       : 770   4   : 900            4       :1526   3       : 412
##  3       : 767   5   :5182            5       : 686   4       : 117
##  (Other):1949   NA's: 913            (Other): 452   5       :  46   ##
NA's    : 136                          NA's    : 375   (Other):  40
##  HHStatus    home_type     Ethnicity    Language
##  1    :3256   1    :5073   7       :5811   1    :7794
##  2    :3670   2    : 655   5       :1231   2    : 579
##  3    :1827   3    :2373   3       : 910   3    : 261
##  NA's: 240   4    : 151   2       : 477   NA's: 359
##               5    : 384   8       : 225
##               NA's: 357   (Other): 271
##                            NA's    :  68

Mode = function(x){
ta = table(x)   tam =
max(ta)   if (all(ta
== tam))     mod = NA
else
    if(is.numeric(x))
      mod = as.numeric(names(ta)[ta == tam])
```

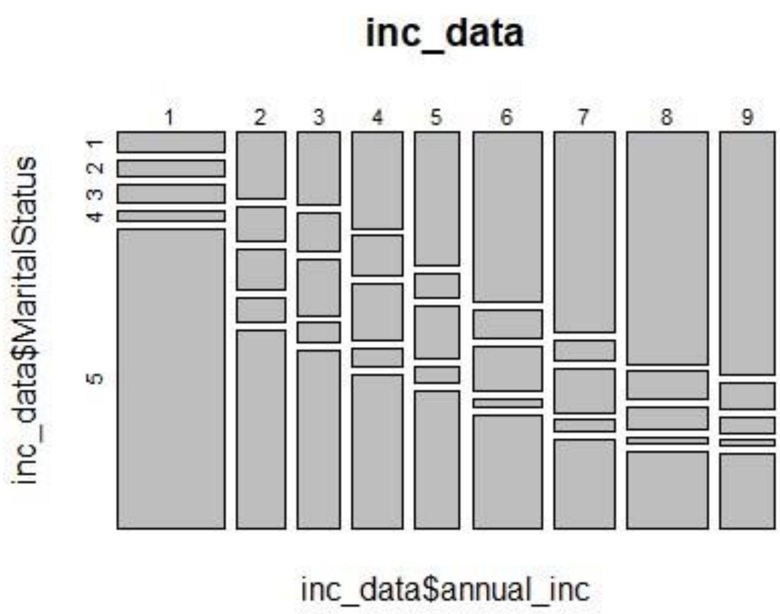
```
  else
    mod = names(ta)[ta == tam]
return(mod)
}

#Imputing the missing values with mode
inc_data$Education[is.na(inc_data$Education)] <- Mode(inc_data$Education)
inc_data$Occupation[is.na(inc_data$Occupation)] <- Mode(inc_data$Occupation)
inc_data$MaritalStatus[is.na(inc_data$MaritalStatus)]<- Mode(inc_data$Marital
Status)
inc_data$Ethnicity[is.na(inc_data$Ethnicity)] <- Mode(inc_data$Ethnicity)
inc_data$Language[is.na(inc_data$Language)] <- Mode(inc_data$Language)
inc_data$home_type[is.na(inc_data$home_type)] <- Mode(inc_data$home_type)
inc_data$ppl_household[is.na(inc_data$ppl_household)] <- Mode(inc_data$ppl_ho
usehold) inc_data$HHStatus[is.na(inc_data$HHStatus)] <-
Mode(inc_data$HHStatus)

library(vcd)

## Loading required package: grid

### PART-A #############################

# Important variables analysed from the plots that follow,
# These are major variables that could be used in prediction
# Age
   # Marital status
   # Education
   # occupation
   # household status
   # home type
   # ethnicity

par(mfrow=c(1,1))
# Increasing trend observed between age and income, that can be winessed
# from high proportion of 3,4,5,6 age groups as we from annual income class 1
to 9
mosaicplot(inc_data$annual_inc ~ inc_data$Age, inc_data)
```



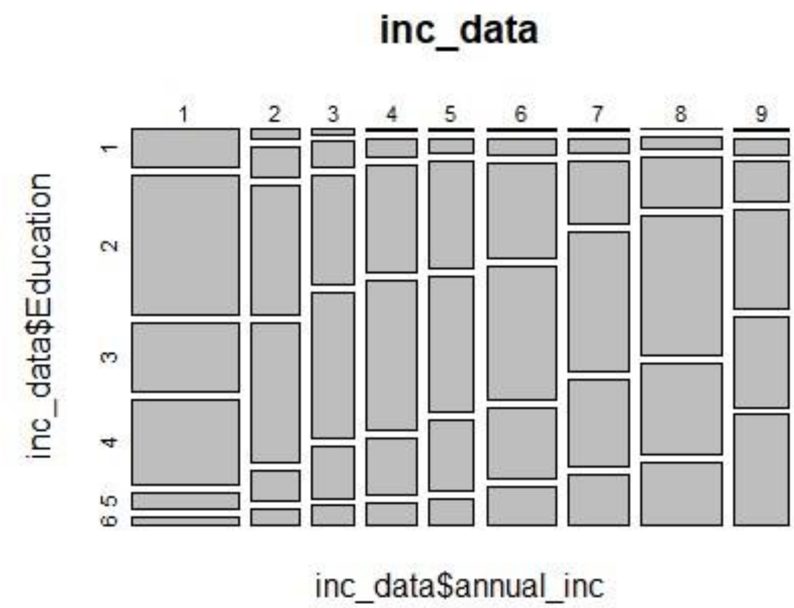

```r
# No significant trend observed between annual income and sex as can be seen from plot
mosaicplot(inc_data$annual_inc ~ inc_data$sex, inc_data)
```



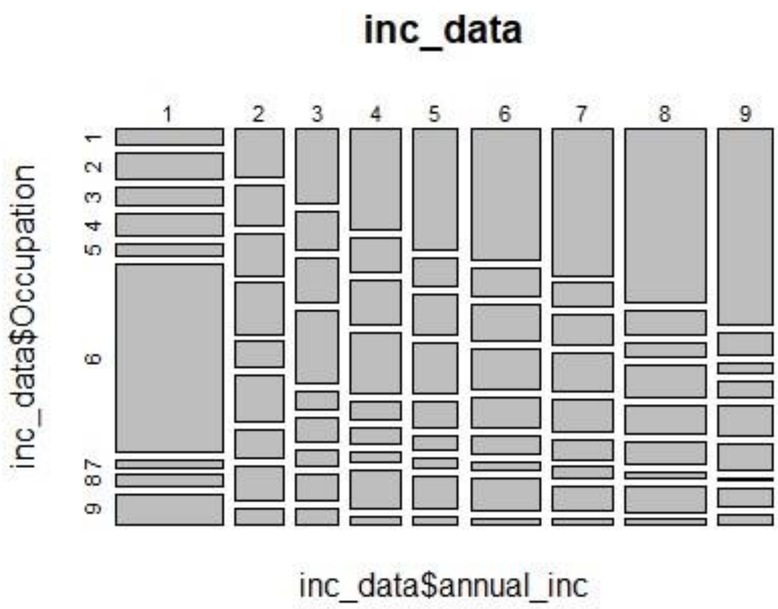
```r
# People who are married or living together (class1 and class2) show positive trend
# with annual income as observed from the below plots
mosaicplot(inc_data$annual_inc ~ inc_data$MaritalStatus, inc_data)
```



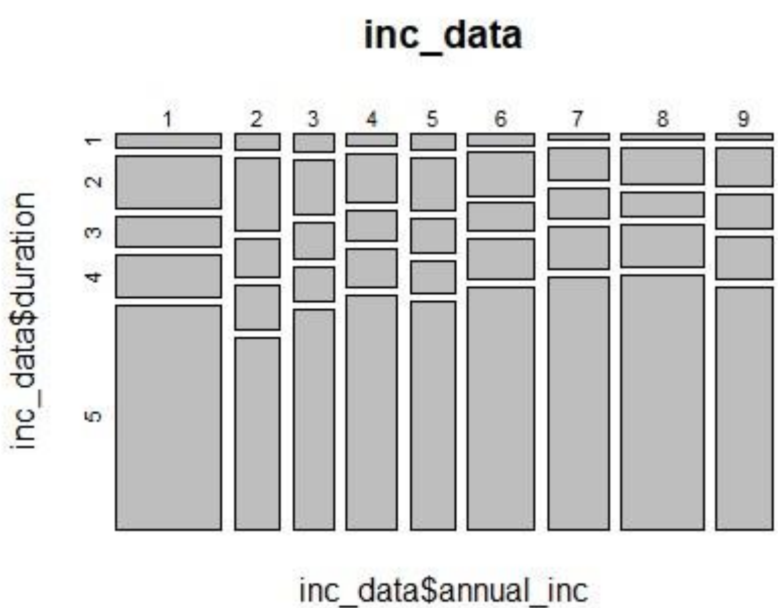

```r
# Education shows an increasing trend with income, higher the education higher the income,
# which can be inferred from the graphs
mosaicplot(inc_data$annual_inc ~ inc_data$Education, inc_data)
```



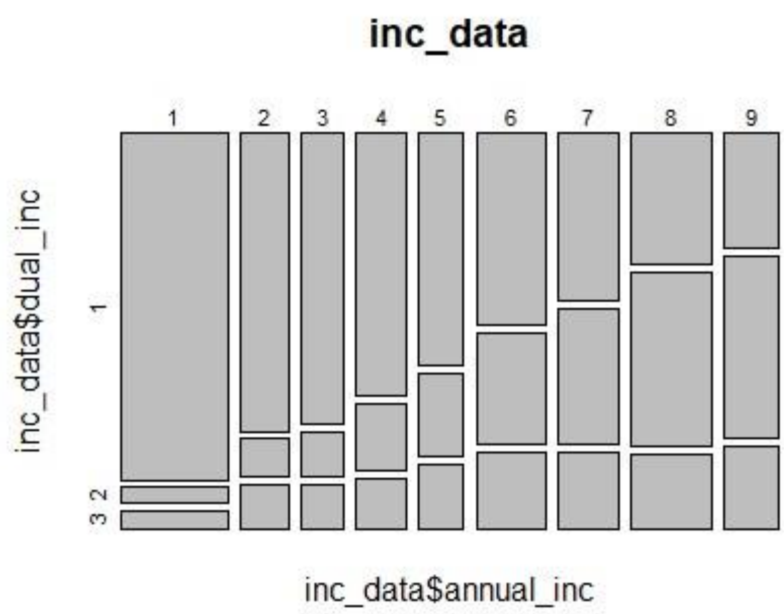

```
#People who belong to occupation(1,2) tend to increase as annual income incre
ases,
# which shows it's an important variable
mosaicplot(inc_data$annual_inc ~ inc_data$Occupation, inc_data)
```



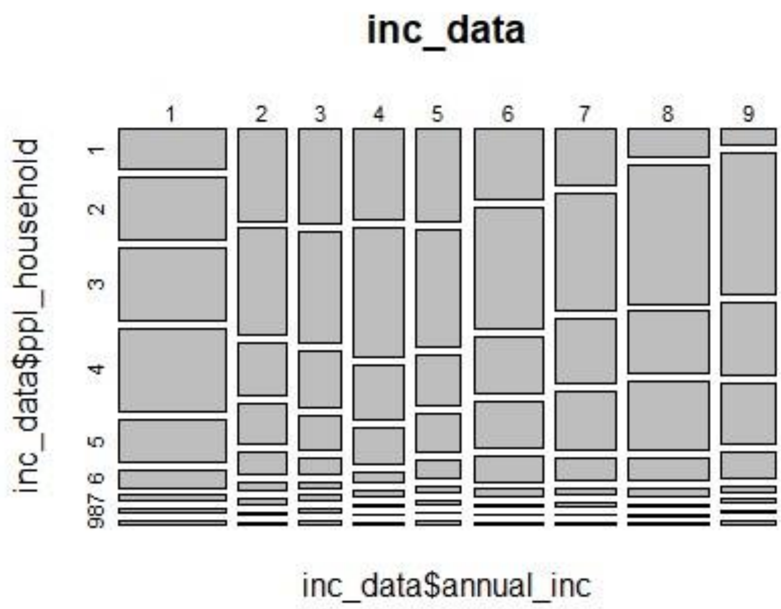

```r
# Duration lived doesn't have any impact on the annual income earned
mosaicplot(inc_data$annual_inc ~ inc_data$duration, inc_data)
```



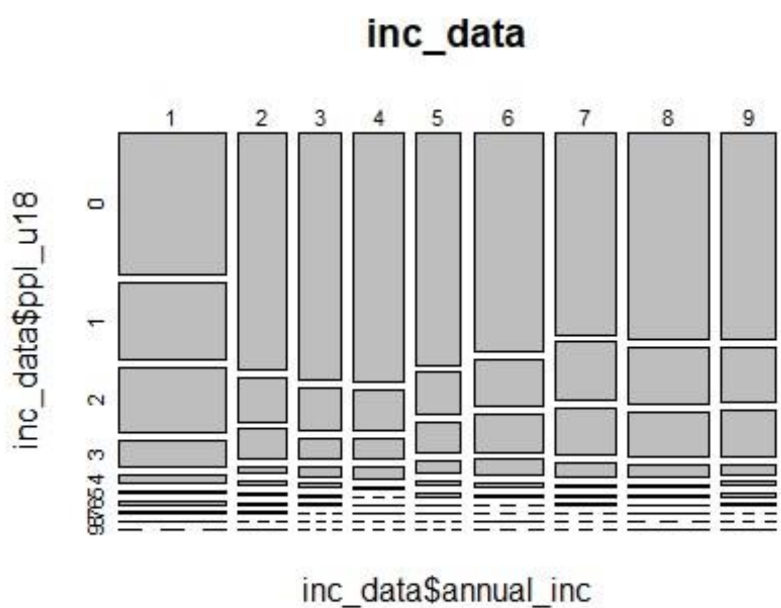
```
#dual_inc (particularly class2) shows positive trend with annual income
mosaicplot(inc_data$annual_inc ~ inc_data$dual_inc, inc_data)
```



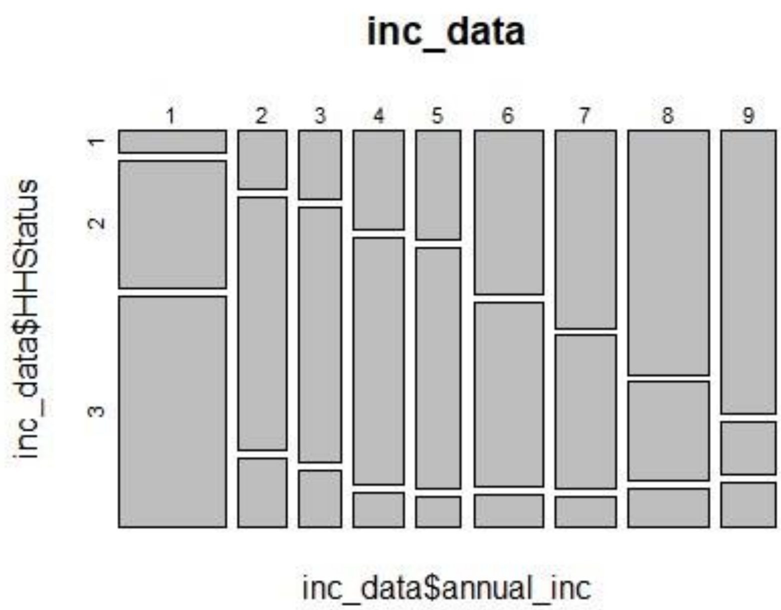

```
# No significant trends are observed between people in house and annual incom
es
mosaicplot(inc_data$annual_inc ~ inc_data$ppl_household, inc_data)
```



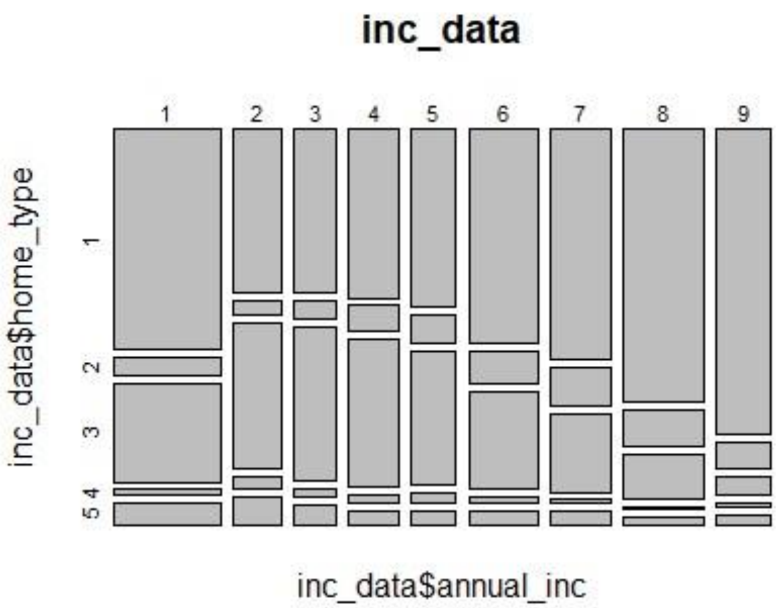

```r
# No significant trends are observed between people under 18 and annual incom
es
mosaicplot(inc_data$annual_inc ~ inc_data$ppl_u18, inc_data)
```



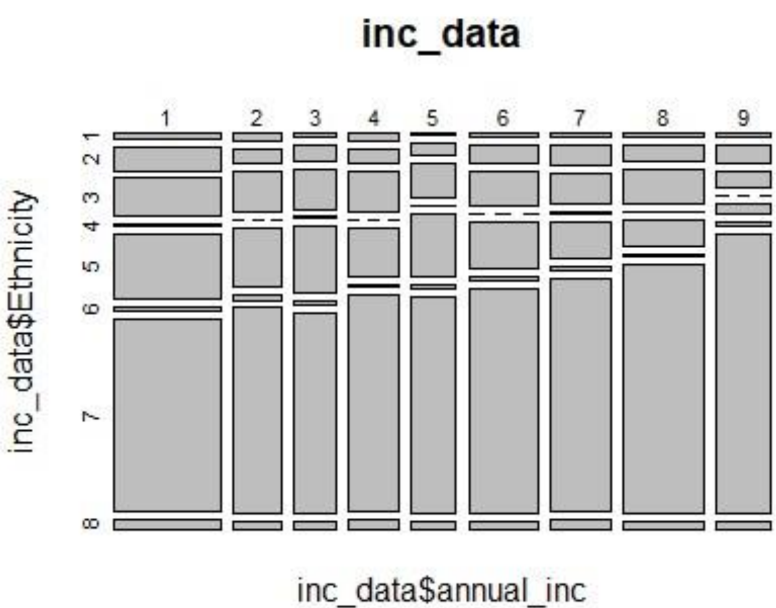
```r
# Having an own house is directly proportional to income,
# which can be witnessed from the plot
mosaicplot(inc_data$annual_inc ~ inc_data$HHStatus, inc_data)
```



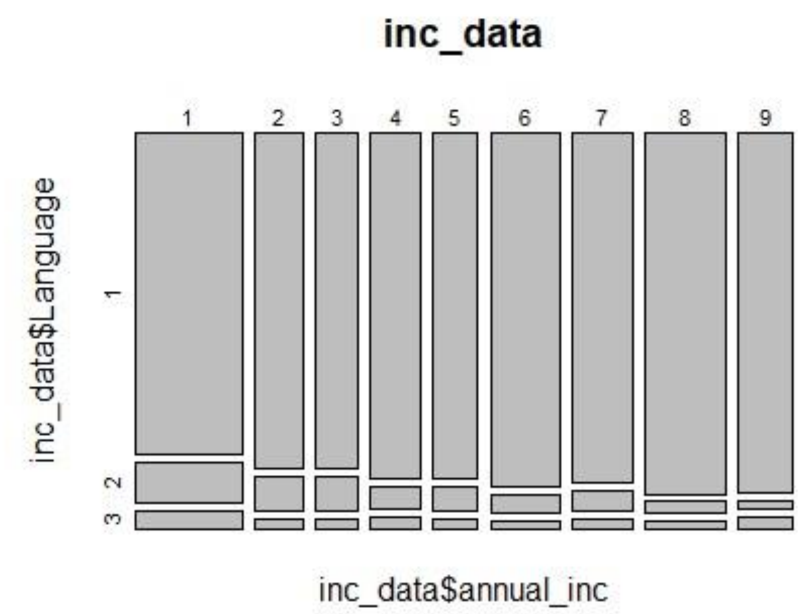

```r
#There is increasing trend in home_type 1 as we from income class 1 to 9
mosaicplot(inc_data$annual_inc ~ inc_data$home_type, inc_data)
```




```r
# White ethnicity shows positive relation with higher incomes,
# but the trend is not as sharp
mosaicplot(inc_data$annual_inc ~ inc_data$Ethnicity, inc_data)
```



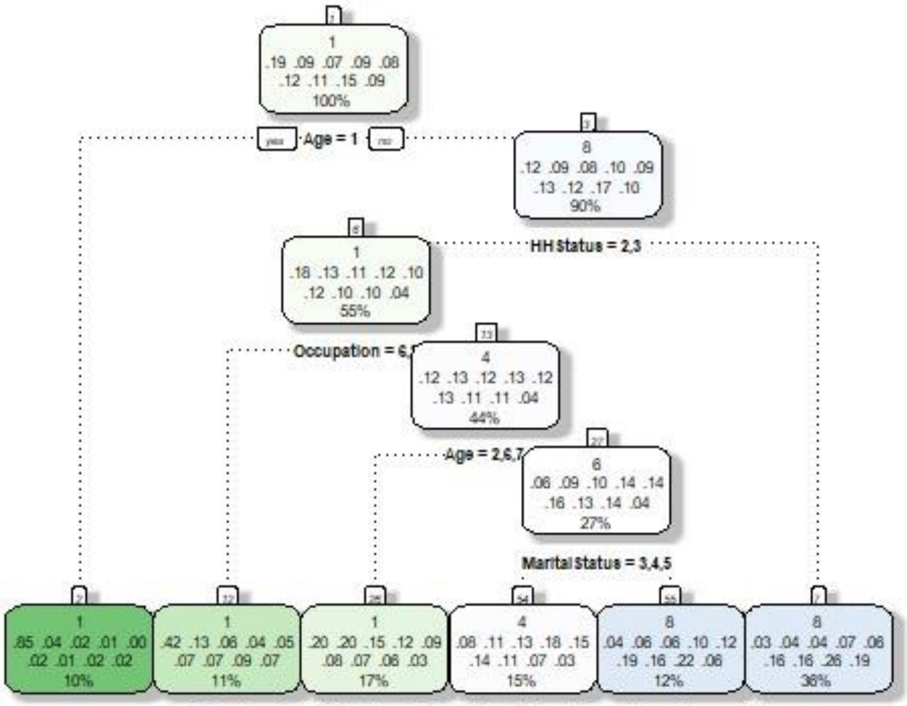
```r
# No significant trends are observed between Language and annual incomes
mosaicplot(inc_data$annual_inc ~ inc_data$Language, inc_data)
```




```r
### PART-B ##############################

# We fix the seed so that every time we run the model we do not work with dif ferent samples set.seed(1234) nrow(inc_data) ## [1] 8993

index = sample(2, nrow(inc_data), replace = TRUE, prob = c(0.6,0.4))

TrainData = inc_data[index == 1, ]

nrow(TrainData)

## [1] 5410
                                   DECISION TREES WITH PARTY PACKAGE**
                                   **********************************

TestData = inc_data[index == 2,]
```

```
nrow(TestData)

## [1] 3583
                                        ages("rpart")
                        tall.packages("
                        tall.packages("
# install.pa
```

```
# install.packages("caret")

library(party)

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

# constructing decision trees using default values and plotting the decision
trees library(rattle)

## Rattle: A free graphical interface for data science with R.
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data. library(caret)

## Loading required package: lattice ##

Loading required package: ggplot2

library(rpart) library(rpart.plot)
library(partykit)

##
## Attaching package: 'partykit'

## The following objects are masked from 'package:party':
##
##     cforest, ctree, ctree_control, edge_simple, mob, mob_control,
##     node_barplot, node_bivplot, node_boxplot, node_inner,
##     node_surv, node_terminal

tree_default = rpart(annual_inc~., data = TrainData,method = "class")
# rpart.plot(tree_default)
fancyRpartPlot(tree_default,tweak=1.5)
```




Rattle 2017-Oct-04 13:19:48 rakesh reddy

```
summary(tree_default)

## Call:
## rpart(formula = annual_inc ~ ., data = TrainData, method = "class") ##
n= 5410
##
##           CP nsplit rel error    xerror        xstd
## 1 0.05201916      0 1.0000000 1.0000000 0.006581136
## 2 0.04928131      1 0.9479808 0.9429614 0.007126201
## 3 0.01688341      2 0.8986995 0.9037189 0.007431329
## 4 0.01140771      4 0.8649327 0.8818161 0.007579994 ##
5 0.01000000      5 0.8535250 0.8674424 0.007669711 ##
## Variable importance
##           Age     Education      HHStatus    Occupation MaritalStatus  ##
46           14            14            10             7
##      dual_inc     home_type ppl_household       ppl_u18  ##
5             3             1             1  ##
## Node number 1: 5410 observations,    complexity param=0.05201916
##   predicted class=1  expected loss=0.8101664  P(node) =1
##     class counts:  1027   478   391   486   435   665   607   820   501
##    probabilities: 0.190 0.088 0.072 0.090 0.080 0.123 0.112 0.152
0.093  ##   left son=2 (522 obs) right son=3 (4888 obs) ##   Primary
splits:
##       Age          splits as  LRRRRRR,   improve=289.9383, (0 missing)
##       Occupation   splits as  RRRRRLRRL, improve=280.1222, (0 missing)
```

```
##       Education     splits as  LLRRRR,    improve=224.2028, (0 missing)
##       HHStatus      splits as  RRL,       improve=212.6706, (0 missing) ##
MaritalStatus splits as  RRRLL,     improve=173.8055, (0 missing) ##
Surrogate splits:
##       Education splits as  LLRRRR,     agree=0.939, adj=0.368, (0 split)
##       ppl_u18   splits as  RRRRRRRRLR, agree=0.904, adj=0.002, (0 split)
##
## Node number 2: 522 observations
##   predicted class=1  expected loss=0.1455939  P(node) =0.09648799
##     class counts:   446    19     8     7     2     9     7    11    13
##    probabilities: 0.854 0.036 0.015 0.013 0.004 0.017 0.013 0.021 0.025
##
## Node number 3: 4888 observations,    complexity param=0.04928131
##   predicted class=8  expected loss=0.8344926  P(node) =0.903512
##     class counts:   581   459   383   479   433   656   600   809   488
##    probabilities: 0.119 0.094 0.078 0.098 0.089 0.134 0.123 0.166
0.100  ##   left son=6 (2961 obs) right son=7 (1927 obs) ##   Primary
splits:
##       HHStatus      splits as  RLL,       improve=102.75400, (0 missing)
##       MaritalStatus splits as  RLLLL,     improve= 98.57541, (0 missing)
##       dual_inc      splits as  LRR,       improve= 88.88052, (0 missing)
##       Age           splits as  -LRRRRR,   improve= 82.41319, (0 missing)
##       Occupation    splits as  RLLLRLLLL, improve= 79.73223, (0 missing)
##   Surrogate splits:
##       MaritalStatus splits as  RLLRL,     agree=0.742, adj=0.345, (0 split
)
##       Age           splits as  -LLRRRR,   agree=0.738, adj=0.334, (0 split
)
##       dual_inc      splits as  LRR,       agree=0.719, adj=0.287, (0 split
)
##       home_type     splits as  RLLRL,     agree=0.693, adj=0.222, (0 split
)
##       Occupation    splits as  LLLLRLLRL, agree=0.664, adj=0.149, (0 split
)
##
## Node number 6: 2961 observations,    complexity param=0.01688341
##   predicted class=1  expected loss=0.8230328  P(node) =0.5473198
##     class counts:   524   385   315   341   309   356   293   308   130
##    probabilities: 0.177 0.130 0.106 0.115 0.104 0.120 0.099 0.104
0.044  ##   left son=12 (596 obs) right son=13 (2365 obs) ##   Primary
splits:
##       Occupation    splits as  RRRRRLRRL, improve=54.86635, (0 missing)
##       Age           splits as  -LRRRRL,   improve=40.50810, (0 missing)
##       MaritalStatus splits as  RLLLL,     improve=25.56575, (0 missing)
##       dual_inc      splits as  LRL,       improve=22.15035, (0 missing)
##       Education     splits as  LLLLRR,    improve=21.28635, (0 missing)
##   Surrogate splits:
##       ppl_u18 splits as  RRRRRRRR-L, agree=0.799, adj=0.002, (0 split) ##
```

```
## Node number 7: 1927 observations
```

```
##   predicted class=8  expected loss=0.7400104  P(node) =0.3561922
##     class counts:    57    74    68   138   124   300   307   501   358
##    probabilities: 0.030 0.038 0.035 0.072 0.064 0.156 0.159 0.260 0.186
##
## Node number 12: 596 observations
##   predicted class=1  expected loss=0.5822148  P(node) =0.1101664
##     class counts:   249    78    35    23    29    43    41    54    44
##    probabilities: 0.418 0.131 0.059 0.039 0.049 0.072 0.069 0.091 0.074
##
## Node number 13: 2365 observations,    complexity param=0.01688341
##   predicted class=4  expected loss=0.8655391  P(node) =0.4371534
##     class counts:   275   307   280   318   280   313   252   254    86
##    probabilities: 0.116 0.130 0.118 0.134 0.118 0.132 0.107 0.107
0.036  ##   left son=26 (903 obs) right son=27 (1462 obs) ##   Primary
splits:
##       Age           splits as  -LRRRLL,   improve=30.68473, (0 missing)
##       Occupation    splits as  RLLLL-LL-, improve=30.60090, (0 missing)
##       Education     splits as  LLLLRR,    improve=22.62861, (0 missing)
##       MaritalStatus splits as  RRLLL,     improve=18.77287, (0 missing) ##
dual_inc      splits as  LRL,       improve=17.78266, (0 missing) ##
Surrogate splits:
##       HHStatus      splits as  -RL,       agree=0.678, adj=0.156, (0 split
)
##       Occupation    splits as  RLRRR-LL-, agree=0.674, adj=0.147, (0 split
)
##       Education     splits as  RLLRRR,    agree=0.644, adj=0.069, (0 split
)
##       MaritalStatus splits as  RRRLR,     agree=0.634, adj=0.042, (0 split
)
##       ppl_household splits as  RRRRRLRLL, agree=0.627, adj=0.022, (0 split
)
##
## Node number 26: 903 observations
##   predicted class=1  expected loss=0.7984496  P(node) =0.1669131
##     class counts:   182   181   137   109    82    72    59    55    26
##    probabilities: 0.202 0.200 0.152 0.121 0.091 0.080 0.065 0.061 0.029
##
## Node number 27: 1462 observations,    complexity param=0.01140771
##   predicted class=6  expected loss=0.8351573  P(node) =0.2702403
##     class counts:    93   126   143   209   198   241   193   199    60
##    probabilities: 0.064 0.086 0.098 0.143 0.135 0.165 0.132 0.136 0.041
##   left son=54 (822 obs) right son=55 (640 obs) ##
Primary splits:
##       MaritalStatus splits as  RRLLL,     improve=16.986890, (0 missing)
##       Occupation    splits as  RRLLL-LL-, improve=15.399760, (0 missing)
##       dual_inc      splits as  LRR,       improve=13.099220, (0 missing)
```

```
##       Education     splits as  LLLLRR,    improve=12.255230, (0 missing)
##       ppl_household splits as  LRLLLLLLL, improve= 6.301725, (0 missing)
##   Surrogate splits:
##       dual_inc      splits as  LRR,        agree=0.902, adj=0.775, (0 spli
```

```
t)
##       ppl_household splits as  LRRRRRRRL,  agree=0.721, adj=0.362, (0 spli
t)
##       ppl_u18       splits as  LRRRRRRR--, agree=0.664, adj=0.233, (0 spli
t)
##       Occupation    splits as  LLLLR-LL-,  agree=0.595, adj=0.075, (0 spli
t)
##       Ethnicity     splits as  RLLRRRLR,   agree=0.591, adj=0.066, (0 spli
t) ##
## Node number 54: 822 observations
##   predicted class=4  expected loss=0.8199513  P(node) =0.1519409
##     class counts:    68    87   107   148   123   118    91    56    24
##    probabilities: 0.083 0.106 0.130 0.180 0.150 0.144 0.111 0.068 0.029
##
## Node number 55: 640 observations
##   predicted class=8  expected loss=0.7765625  P(node) =0.1182994
##     class counts:    25    39    36    61    75   123   102   143    36
##    probabilities: 0.039 0.061 0.056 0.095 0.117 0.192 0.159 0.223 0.056
```

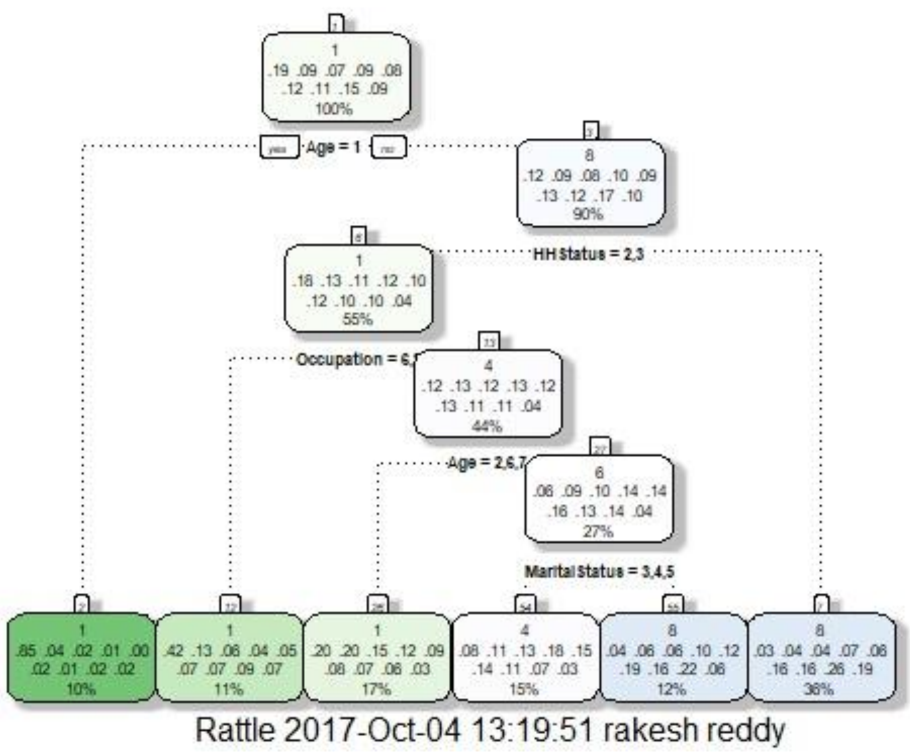
```
tree_default1 <- as.party(tree_default) nodeids(tree_default1,terminal
= TRUE)
```

```
## [1]  2  5  7  9 10 11
```

```
#Number of leaves in the system are 6 as can be seen from plot
#Size of leaf node below as follows #
Node number 2: 522 observations
# Node number 7: 1927 observations
# Node number 12: 596 observations
# Node number 26: 903 observations
# Node number 54: 822 observations
# Node number 55: 640 observations

### PART-C #############################

# Age,Education,Occupation,HHStatus,MaritalStatus are important variables fro
m the tree
# as suggested by variable importance and primary splits

# Variable importance
# Age      Education       HHStatus     Occupation MaritalStatus       dual_inc
# 46             14             14             10             7             5
# home_type ppl_household       ppl_u18
# 3             1             1
```

```
# Primary splits:
#   Age           splits as  LRRRRRR,   improve=289.9383, (0 missing) #
Occupation    splits as  RRRRRLRRL, improve=280.1222, (0 missing)
# Education     splits as  LLRRRR,    improve=224.2028, (0 missing)
```

```
# HHStatus      splits as  RRL,       improve=212.6706, (0 missing)
# MaritalStatus splits as  RRRLL,     improve=173.8055, (0 missing)

# These variables show high similarity with the variables that were obtained
in part A),
# in terms of trends with income, all these variables had a significant trend
.

# Look at the two-way table to check the performance of the mdoel on train da
ta
a <- table(predict(tree_default,type = "class"), TrainData$annual_inc, dnn =
c("predicted", "actual"))  #Compare Predicted vs Actual
error_rate <- 1-sum(diag(a))/sum(a)
error_rate

## [1] 0.6914972

# Accuracy of the default model= (877+148+644)/5410 = 30.8%
# prediction of the model is below par, with almost 70% error on the training
data



### PART-D ##############################
fancyRpartPlot(tree_default,tweak=1.5)
```



```
# Rule1: Person who is living in his own house (HH status=Own) is likely to e
arn higher income
# Explanation: As can be seen from the leaf node 7 (for which support is 36%)
# the confidence of higher income groups (6,7,8,9) : 0.16,0.16,0.26,0.19,  #
this suggests there is a likelihood of 76% to earn more than30,000 per year
if you own a house

# Rule2: If a person who belongs to age group (2,6,7) is likely to earn low i
ncome per year, i.e.,
# Young adults(18-24) and senior citizens(55 and more) are probable to earn l
ow annual income
# Explanation: For Node 26 (for which support is 17%),
# the confidence of lower income groups (1,2,3,4): 0.20,0.20,0.15,0.12
# this suggests there is a likelihood of 67% to earn less than 30,000 per yea
r
# if you belong to these age groups

### PART-E ##############################
# Surrogate splits were not used in the construction of the tree,
# but CART in default provides the surrogate splits

# Meaning of surrogate:

# The ideal surrogate splits the data in exactly the same way as the primary
split,
# in other words, we are looking for clones, close approximations,
# something else in the data that can do the same work that the  #
primary splitter accomplished.
# Surrogates have two primary functions: first, to split the data when  #
the primary splitter is missing.
# Now, the primary splitter may never have been missing in the training data.
# However, when it comes time to make predictions on future data,
# we have no idea whether that particular splitter will always be available.
# When it is missing, then the surrogates will be able to take over
# and take on the work that the primary splitter accomplished during the
# initial building of the tree. In addition, surrogates reveal common pattern
s
# among predictors and the data set.

# Example of surrogate split in the tree constructed here:

# Node number 1: 5410 observations,    complexity param=0.05201916 #
Surrogate splits:
#   Education splits as  LLRRRR,     agree=0.939, adj=0.368, (0 split)
```

```
# ppl_u18   splits as  RRRRRRRRLR, agree=0.904, adj=0.002, (0 split)
```

```
### PART-F ##############################

# confusion matrix for test error prediction
b <- table(predict(tree_default,TestData,type = "class"), TestData$annual_inc
, dnn = c("predicted", "actual"))
error_b <- 1-sum(diag(b))/sum(b) error_b


## [1] 0.6943902
                          del test data = (615+0+0+91+0+0+0+389+0)/3583 =
#
# so, test error is 69.5%

### PART-                 ########################
                          /Profile of high income groups
                          he path of node 1-3-7 on extreme right (p of
#                         ;ives out the below profile of high income
# Age group: Above 18
# Household status : Own


                          he path of nodes 1-3-6-13-27-55 (p of 55 node:17%)
#    Occupation:1,2,3,4,5,7he below profile of high income
Age:3,4,5 (from 25            groups cept student and
                              unemployed) -54)
                          :1,2 (Married or living together)

# Combining both these paths info, profile should look as below:

#   Age Group: 25-54

#   Household status: Own

#   Occupation:Some Sort of employment

#   Marital status : Married or living together
```

```
### PART-H ############################## inc_big
<- read_excel( nt 2/income.big.xlsx",                    =Desktop/Business
                          col_names    =    FALSE,    na    g/Assignme
summary(inc_big)
                                                             "NA")
##       X__1            X__2            X__3
##  Min.   :1.000   Min.   :1.000   Min.   :1.00   Min.
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:1.00   1st Q
##  Median :5.000   Median :2.000   Median :3.00
##  Mean   :4.867   Mean   :1.546   Mean   :3.02   Mean
##  3rd Qu.:7.000   3rd Qu.:2.000   3rd Qu.:5.00   3rd Q
##  Max.   :9.000   Max.   :2.000   Max.   :5.00   Max.     Median
##                                                          :3.000
##       X__5            X__6            X__7
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.
##  1st Qu.:3.000   1st Qu.:1.000   1st Qu.:4.000   1st
##  Median :4.000   Median :4.000   Median :5.000   Medi
```

```
##  Mean   :3.817   Mean   :3.771   Mean   :4.203   Mean   :1.565
##  3rd Qu.:5.000   3rd Qu.:6.000   3rd Qu.:5.000   3rd Qu.:2.000
##  Max.   :6.000   Max.   :9.000   Max.   :5.000   Max.   :3.000
##  NA's   :53      NA's   :94      NA's   :720
##       X__9           X__10             X__11           X__12
##  Min.   :1.000   Min.   :0.0000   Min.   :1.00   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:0.0000   1st Qu.:1.00   1st Qu.:1.000
##  Median :3.000   Median :0.0000   Median :2.00   Median :1.000
##  Mean   :2.897   Mean   :0.6862   Mean   :1.83   Mean   :1.831
##  3rd Qu.:4.000   3rd Qu.:1.0000   3rd Qu.:2.00   3rd Qu.:3.000
##  Max.   :9.000   Max.   :9.0000   Max.   :3.00   Max.   :5.000
##  NA's   :261                      NA's   :183    NA's   :263
##      X__13           X__14           X__15           X__16
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:5.000   1st Qu.:1.000   1st Qu.:3.000   1st Qu.:3.000
##  Median :7.000   Median :1.000   Median :5.000   Median :5.000
##  Mean   :5.953   Mean   :1.129   Mean   :4.989   Mean   :4.977
##  3rd Qu.:7.000   3rd Qu.:1.000   3rd Qu.:7.000   3rd Qu.:7.000
##  Max.   :8.000   Max.   :3.000   Max.   :9.000   Max.   :9.000
##  NA's   :55      NA's   :198                     NA's   :1
##      X__17           X__18           X__19           X__20
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.00
##  1st Qu.:3.000   1st Qu.:3.000   1st Qu.:3.000   1st Qu.:3.00
##  Median :5.000   Median :5.000   Median :5.000   Median :5.00
##  Mean   :5.009   Mean   :4.961   Mean   :5.035   Mean   :4.97
##  3rd Qu.:7.000   3rd Qu.:7.000   3rd Qu.:7.000   3rd Qu.:7.00
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.00
##  NA's   :1       NA's   :1       NA's   :1       NA's   :1
##      X__21           X__22       X__23           X__24
##  Min.   :1.000   Min.   :1   Min.   :1.000   Min.   :1.00
##  1st Qu.:3.000   1st Qu.:3   1st Qu.:3.000   1st Qu.:3.00
##  Median :5.000   Median :5   Median :5.000   Median :5.00
##  Mean   :5.017   Mean   :5   Mean   :5.004   Mean   :5.05
##  3rd Qu.:7.000   3rd Qu.:7   3rd Qu.:7.000   3rd Qu.:7.00
##  Max.   :9.000   Max.   :9   Max.   :9.000   Max.   :9.00   ##
NA's   :1       NA's   :1   NA's   :1       NA's   :1

colnames(inc_big) <- c('annual_inc',
                        'sex', 'MaritalStatus',
                        'Age', 'Education', 'Occupation',
                        'duration','dual_inc',
                        'ppl_household','ppl_u18','HHStatus',
                        'home_type','Ethnicity','Language','spur_1','spur_2',
'spur_3',
                        'spur_4','spur_5','spur_6','spur_7','spur_8','spur_9'
,'spur_10')
```

```r
colSums(is.na(inc_big))
```

```
##    annual_inc           sex MaritalStatus           Age     Education
##             0             0           101             0            53
##    Occupation      duration      dual_inc ppl_household       ppl_u18
##            94           720             0           261             0
##      HHStatus     home_type     Ethnicity      Language        spur_1
##           183           263            55           198             0
##        spur_2        spur_3        spur_4        spur_5        spur_6  ##
1             1             1             1             1
##        spur_7        spur_8        spur_9       spur_10  ##
1             1             1             1
```

```
inc_big <- as.data.frame(lapply(inc_big,factor)) str(inc_big)
```

```
## 'data.frame':    6508 obs. of  24 variables:
##  $ annual_inc   : Factor w/ 9 levels "1","2","3","4",..: 9 9 9 1 1 8 1 6 2
4 ...
##  $ sex          : Factor w/ 2 levels "1","2": 2 1 2 2 2 1 1 1 1 1 ... ##
$ MaritalStatus: Factor w/ 5 levels "1","2","3","4",..: 1 1 1 5 5 1 5 3 1 1
...
##  $ Age          : Factor w/ 7 levels "1","2","3","4",..: 5 5 3 1 1 6 2 3 6
7 ...
##  $ Education    : Factor w/ 6 levels "1","2","3","4",..: 4 5 5 2 2 4 3 4 3
4 ...
##  $ Occupation   : Factor w/ 9 levels "1","2","3","4",..: 5 5 1 6 6 8 9 3 8
8 ...
##  $ duration     : Factor w/ 5 levels "1","2","3","4",..: 5 5 5 5 3 5 4 5 5
4 ...
##  $ dual_inc     : Factor w/ 3 levels "1","2","3": 3 3 2 1 1 3 1 1 3 3 ...
##  $ ppl_household: Factor w/ 9 levels "1","2","3","4",..: 3 5 3 4 4 2 3 1 3
2 ...
##  $ ppl_u18      : Factor w/ 10 levels "0","1","2","3",..: 1 3 2 3 3 1 2 1
1 1 ...
##  $ HHStatus     : Factor w/ 3 levels "1","2","3": 1 1 2 3 3 1 2 2 2 2 ...
##  $ home_type    : Factor w/ 5 levels "1","2","3","4",..: 1 1 3 1 1 1 3 3 3
3 ...
##  $ Ethnicity    : Factor w/ 8 levels "1","2","3","4",..: 7 7 7 7 7 7 7 7 7
7 ...
##  $ Language     : Factor w/ 3 levels "1","2","3": NA 1 1 1 1 1 1 1 1 1 ...
##  $ spur_1       : Factor w/ 9 levels "1","2","3","4",..: 4 4 5 9 8 5 2 5 9
2 ...
##  $ spur_2       : Factor w/ 9 levels "1","2","3","4",..: 5 8 1 3 1 2 7 2 8
8 ...
##  $ spur_3       : Factor w/ 9 levels "1","2","3","4",..: 2 9 1 7 9 9 5 7 9
6 ...
##  $ spur_4       : Factor w/ 9 levels "1","2","3","4",..: 5 8 7 6 9 9 3 7 7
8 ...
##  $ spur_5       : Factor w/ 9 levels "1","2","3","4",..: 9 3 7 9 8 7 4 9 4
9 ...
##  $ spur_6       : Factor w/ 9 levels "1","2","3","4",..: 6 8 9 5 8 3 2 2 3
```

```
2 ...
##  $ spur_7       : Factor w/ 9 levels "1","2","3","4",..: 3 7 6 4 1 5 8 8 1
9 ...
##  $ spur_8       : Factor w/ 9 levels "1","2","3","4",..: 7 2 6 2 3 8 8 8 1
7 ...
##  $ spur_9       : Factor w/ 9 levels "1","2","3","4",..: 9 7 7 9 8 5 2 5 3
8 ...
##  $ spur_10      : Factor w/ 9 levels "1","2","3","4",..: 6 6 2 3 5 1 5 4 3
9 ...
```

```
inc_big$sex <- factor(inc_big$sex,c('1','2'),c('Male','Female'))

# treating missing values with mode

for(i in 1:ncol(inc_big)){
  inc_big[is.na(inc_big[,i]), i] <- mode(inc_big[,i])
}
```

```
## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated
```

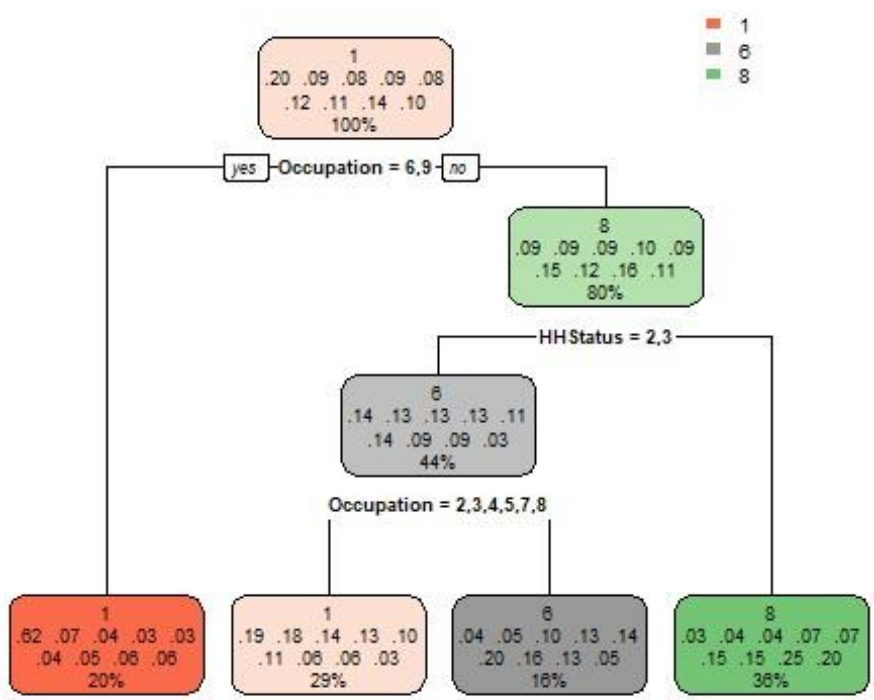
```
## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
```

```
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

## Warning in `[<-.factor`(`*tmp*`, iseq, value = c("numeric", "numeric",
## "numeric", : invalid factor level, NA generated

# checking if missing values are treated colSums(is.na(inc_big))

##    annual_inc           sex MaritalStatus           Age     Education
##             0             0           101             0            53
##    Occupation      duration      dual_inc ppl_household       ppl_u18
##            94           720             0           261             0
```

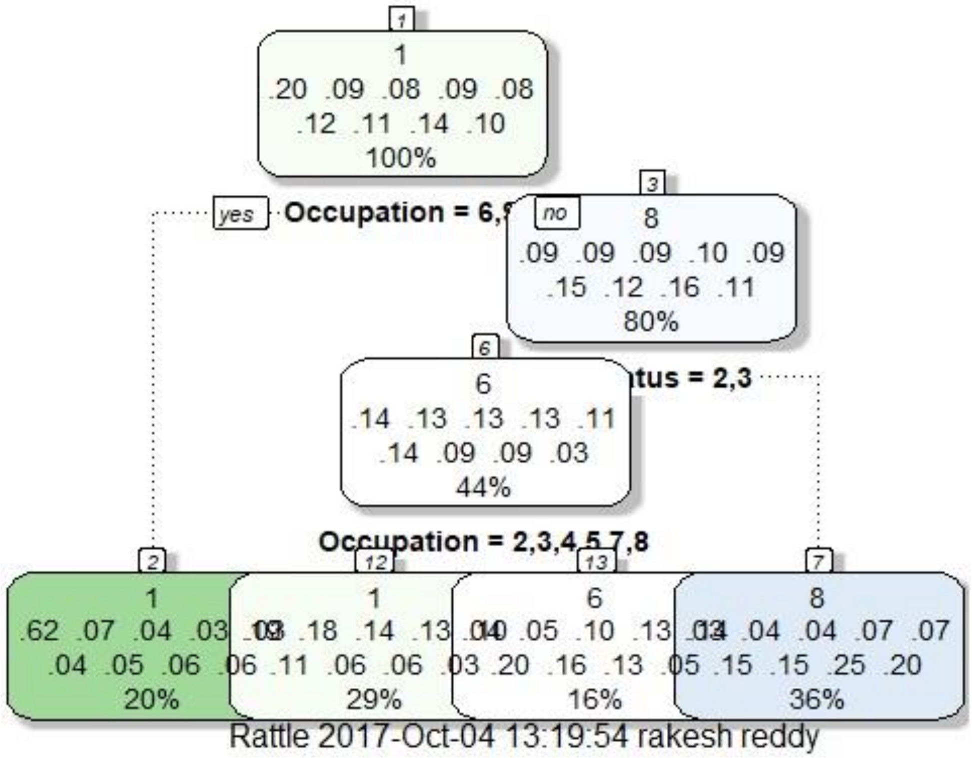
```
##       HHStatus      home_type      Ethnicity       Language         spur_1
```

```
##          183          263           55          198            0
##       spur_2       spur_3       spur_4       spur_5       spur_6
##            1            1            1            1            1
##       spur_7       spur_8       spur_9      spur_10
##            1            1            1            1
```

```
# constructing tree for income_big by taking training data with same  #
number of observations in training data of part(c) ~ 5410 samples
index = sample(2, nrow(inc_big), replace = TRUE, prob = c(0.835,0.165))
TrainData_big = inc_big[index == 1, ] nrow(TrainData_big)

## [1] 5448

TestData_big = inc_big[index == 2,] nrow(TestData_big) ## [1] 1060
big_tree = rpart(annual_inc~., data = TrainData_big,method = "class")

rpart.plot(big_tree)
```




```
fancyRpartPlot(big_tree,tweak=1.5)
```




```
summary(big_tree)

## Call:
## rpart(formula = annual_inc ~ ., data = TrainData_big, method = "class") ##
n= 5448
##
##            CP nsplit rel error    xerror        xstd
## 1 0.07366021      0 1.0000000 1.0000000 0.006670575
## 2 0.02930445      1 0.9263398 0.9263398 0.007331004 ##
3 0.01000000      3 0.8677309 0.8716078 0.007702247 ##
## Variable importance
##    Occupation      HHStatus           Age     Education MaritalStatus  ##
45            18            16             6             5
##      dual_inc     home_type
##             5             5
##
## Node number 1: 5448 observations,    complexity param=0.07366021
##   predicted class=1  expected loss=0.8048825  P(node) =1
##     class counts:  1063   469   435   478   447   676   575   776   529
##    probabilities: 0.195 0.086 0.080 0.088 0.082 0.124 0.106 0.142
0.097  ##   left son=2 (1084 obs) right son=3 (4364 obs) ##   Primary
splits:
##       Occupation    splits as  RRRRRLRRL, improve=276.9256, (73 missing)
##      Age           splits as  LRRRRRR,   improve=267.3286, (0 missing)
##      HHStatus      splits as  RRL,       improve=226.0268, (163 missing)
##      Education     splits as  LLRRRR,    improve=214.2599, (47 missing)
```

```
##       MaritalStatus splits as  RRRLL,     improve=167.6487, (84 missing)
##   Surrogate splits:
##       Age       splits as  LRRRRRR,    agree=0.854, adj=0.271, (73 split)
##       Education splits as  LLRRRR,     agree=0.826, adj=0.130, (0 split)
##       HHStatus  splits as  RRL,        agree=0.826, adj=0.129, (0 split)
##       ppl_u18   splits as  RRRRRRRLRR, agree=0.800, adj=0.001, (0 split)
##
## Node number 2: 1084 observations
##   predicted class=1  expected loss=0.3791513  P(node) =0.1989721
##     class counts:   673    77    47    35    33    43    51    63    62
##    probabilities: 0.621 0.071 0.043 0.032 0.030 0.040 0.047 0.058 0.057
##
## Node number 3: 4364 observations,    complexity param=0.02930445
##   predicted class=8  expected loss=0.8366178  P(node) =0.8010279
##     class counts:   390   392   388   443   414   633   524   713   467
##    probabilities: 0.089 0.090 0.089 0.102 0.095 0.145 0.120 0.163
0.107  ##   left son=6 (2402 obs) right son=7 (1962 obs) ##   Primary
splits:
##       HHStatus      splits as  RLL,       improve=97.02602, (139 missing)
##       MaritalStatus splits as  RLLLL,     improve=89.19806, (66 missing)
##       dual_inc      splits as  LRR,       improve=86.24064, (0 missing)
##       Age           splits as  LLRRRRR,   improve=66.01355, (0 missing) ##
Occupation    splits as  RLLLL-LL-, improve=56.80886, (64 missing) ##
Surrogate splits:
##       Age           splits as  LLLRRRR,   agree=0.731, adj=0.405, (139
spl it)
##       MaritalStatus splits as  RLLRL,     agree=0.725, adj=0.392, (0 split
)
##       dual_inc      splits as  LRR,       agree=0.713, adj=0.364, (0 split
)
##       home_type     splits as  RRLRL,     agree=0.710, adj=0.358, (0 split
)
##       Occupation    splits as  RLLLR-LR-, agree=0.624, adj=0.167, (0 split
)
##
## Node number 6: 2402 observations,    complexity param=0.02930445
##   predicted class=6  expected loss=0.8597002  P(node) =0.4408957
##     class counts:   333   320   302   311   275   337   227   213    84
##    probabilities: 0.139 0.133 0.126 0.129 0.114 0.140 0.095 0.089
0.035  ##   left son=12 (1554 obs) right son=13 (848 obs) ##   Primary
splits:
##       Occupation splits as  RLLLL-LL-, improve=37.32463, (40 missing)
##       Age        splits as  LLRRRLL,   improve=35.50909, (0 missing)
##       Education  splits as  LLLRRR,    improve=24.31828, (26 missing)
##       HHStatus   splits as  -RL,       improve=23.83769, (86 missing)
##        dual_inc   splits as  LRL,       improve=17.19950, (0 missing)
##   Surrogate splits:
##       Education splits as  LLLLRR, agree=0.722, adj=0.22, (37 split) ##
```

```
## Node number 7: 1962 observations
```

```
##   predicted class=8  expected loss=0.745158  P(node) =0.3601322
##     class counts:    57    72    86   132   139   296   297   500   383
##    probabilities: 0.029 0.037 0.044 0.067 0.071 0.151 0.151 0.255 0.195
##
## Node number 12: 1554 observations
##   predicted class=1  expected loss=0.8056628  P(node) =0.2852423
##     class counts:   302   278   220   199   153   169    95    99    39
##    probabilities: 0.194 0.179 0.142 0.128 0.098 0.109 0.061 0.064 0.025
##
## Node number 13: 848 observations
##   predicted class=6  expected loss=0.8018868  P(node) =0.1556535
##     class counts:    31    42    82   112   122   168   132   114    45 ##
probabilities: 0.037 0.050 0.097 0.132 0.144 0.198 0.156 0.134 0.053
```

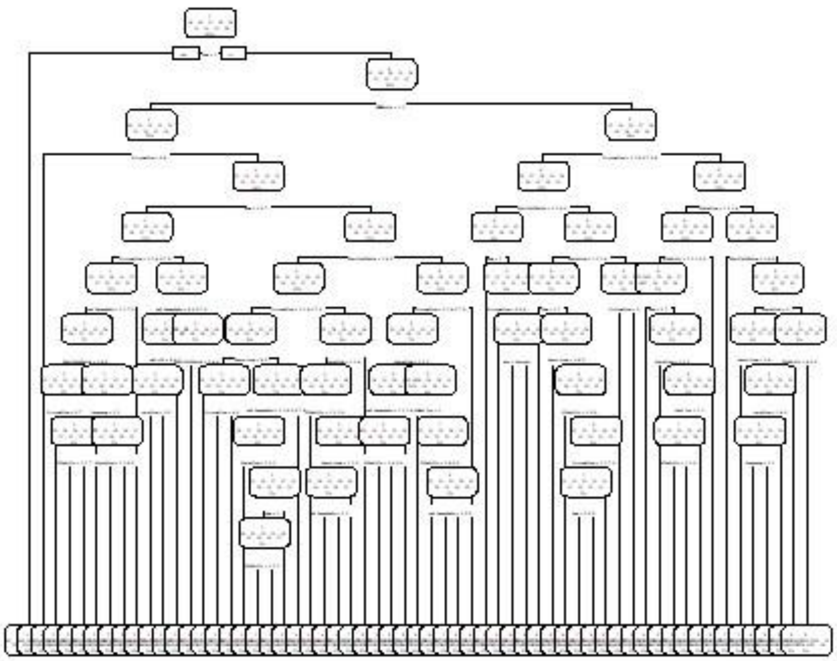
```
# calculating training error for income_big
table(predict(big_tree,type = "class"), TrainData_big$annual_inc, dnn = c("pr
edicted", "actual"))  #Compare Predicted vs Actual
```

```
##          actual
## predicted   1   2   3   4   5   6   7   8   9
##         1 975 355 267 234 186 212 146 162 101
##         2   0   0   0   0   0   0   0   0   0
##         3   0   0   0   0   0   0   0   0   0
##         4   0   0   0   0   0   0   0   0   0
##         5   0   0   0   0   0   0   0   0   0
##         6  31  42  82 112 122 168 132 114  45
##         7   0   0   0   0   0   0   0   0   0
##         8  57  72  86 132 139 296 297 500 383
##         9   0   0   0   0   0   0   0   0   0
```

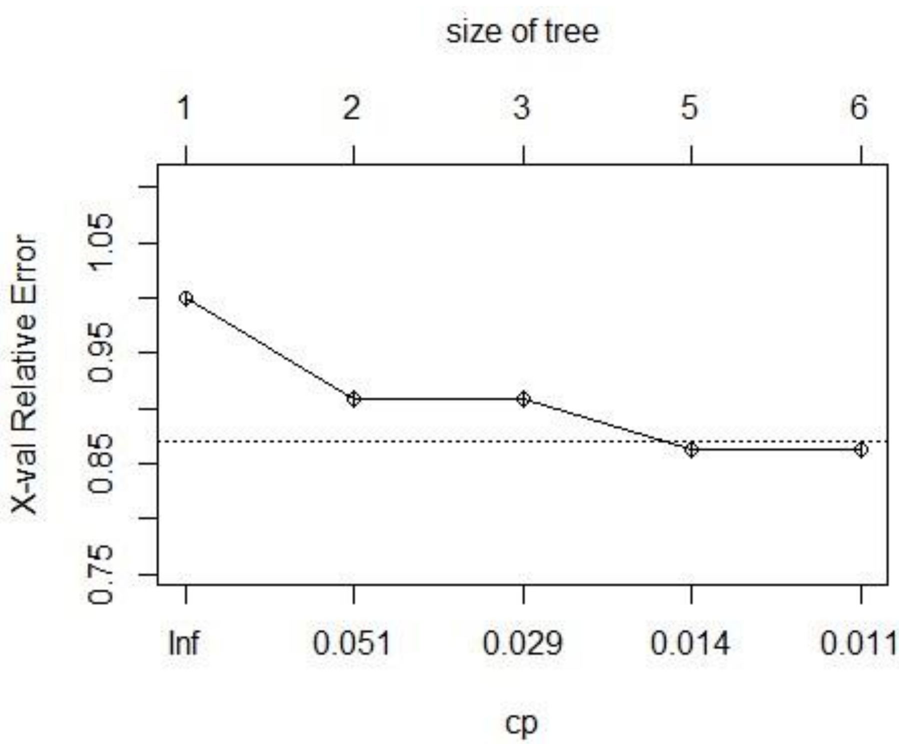
```
# accuracy of the training model income_big=(925+252+509)/5437 = 31.09% (more
or less same as c)
# Accuracy of the default model in part(c)= (877+148+644)/5410 = 30.8%

# Variable importance training model income_big
# Occupation      HHStatus           Age      Education MaritalStatus      home
_type      dual_inc
# 38             22             19             6             6             4
4

# Variable importance for default model in part(c)
# Age     Education      HHStatus     Occupation MaritalStatus      dual_inc
# 46             14             14            10             7             5
# home_type ppl_household   ppl_u18
# 3              1             1

# Top 5 important variables remain same,
# but the order of splitting and importance is different in these model
```

```
### PART-I ##############################

for (i in 1:8){
indec = sample(2, nrow(inc_data), replace = TRUE, prob = c(0.6,0.4))
train_data = inc_data[indec == 1,] test_data = inc_data[indec == 2,]
# assign(paste0("data",i),train_data) #
assign(paste0("data_t",i),test_data)
tree_def = rpart(annual_inc~., data = train_data,method = "class") temp <-
table(predict(tree_def,type = "class"), train_data$annual_inc, dnn =
c("predicted", "actual"))
test_temp <- table(predict(tree_def,test_data,type = "class"), test_data$annu
al_inc, dnn = c("predicted", "actual"))
assign(paste0("training_error_rate",i),1-sum(diag(temp))/sum(temp))
assign(paste0("testing_error_rate",i),1-sum(diag(test_temp))/sum(test_temp))
}

error_matrix <- matrix(c(testing_error_rate1,training_error_rate1,testing_err
or_rate2,training_error_rate2,
                         testing_error_rate3,training_error_rate3,testing_err
or_rate4,training_error_rate4,
                         testing_error_rate5,training_error_rate5,testing_err
or_rate6,training_error_rate6,
                         testing_error_rate7,training_error_rate7,testing_err
or_rate8,training_error_rate8),ncol=2,byrow = T)

colnames(error_matrix) <- c("test","train")


error_matrix


##           test     train
## [1,] 0.6894131 0.6944035
## [2,] 0.7019151 0.6842301



## [7,] 0.6945221 0.6908587
## [8,] 0.6920483 0.6893633
## [3,] 0.6984522 0.6863256
## [4,] 0.6929922 0.6901983
## [5,] 0.6839176 0.6960548
## [6,] 0.6957251 0.6900628


#   All samples showed the similar error rates with test and train samples with
```

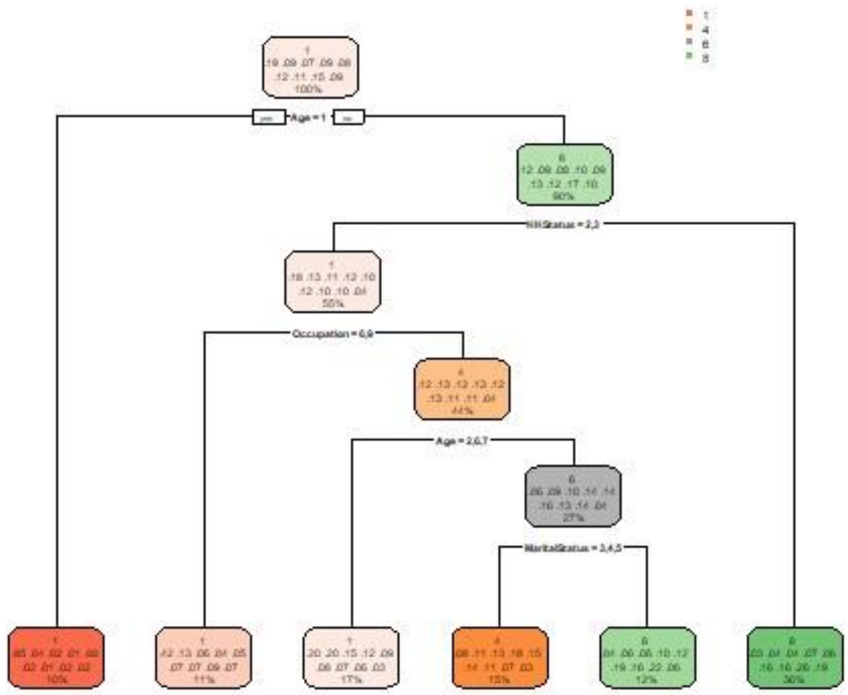
```
tolerance arounf 1-3% from the above matrix,

# which is also expected because of the random sampling method that is chosen




### PART-J ##############################

# constructing a minimum pruning tree
set.seed(1234)
indec = sample(2, nrow(inc_data), replace = TRUE, prob = c(0.6,0.4))
train_data_min_prun = inc_data[indec == 1,] test_data_min_prun =
inc_data[indec == 2,]
# using cp value of 0.001 and constructing the tree
tree_def_min_prun = rpart(annual_inc~., data = train_data_min_prun,method = "
class",cp=0.001) library("partykit")
# summary(tree_def_min_prun) tree_def_min_prun_kt <-
as.party(tree_def_min_prun) #Number of terminal nodes
in minimum pruning tree = 59
nodeids(tree_def_min_prun_kt,terminal = TRUE)

##  [1]   2   5  11  13  14  16  18  19  20  24  25  26  28  29  34  35  38
## [18]  41  42  43  44  47  50  51  52  53  58  59  60  62  64  66  67  68 ##
[35]  72  74  76  77  80  82  84  87  88  89  91  92  96  98 101 102 103
## [52] 104 106 109 112 113 114 116 117 rpart.plot(tree_def_min_prun)

## Warning: All boxes will be white (the box.palette argument will be ignored
) because
## the number of classes predicted by the model 9 is greater than length(box.
palette) 6.
## To make this warning go away use box.palette=0 or trace=-1.

## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



```r
# constructing second tree with leaf node limit=100 and parent node limit=500
tree_def_split = rpart(annual_inc~., data = train_data_min_prun,method = "cla
ss",minsplit=100,minbucket=500) # choosing the best cp from the plot
plotcp(tree_def_split)
```



```
tree_def_split = rpart(annual_inc~., data = train_data_min_prun,method = "cla
ss",minsplit=100,minbucket=500,cp=0.01)
rpart.plot(tree_def_split)
```



```r
tree_def_split_kt <- as.party(tree_def_split) #Number
of terminal nodes in this tree = 4
nodeids(tree_def_split_kt,terminal = TRUE)
```

```
## [1]  2  5  7  9 10 11
```

```r
# Comparing errors of both models pred_train_min_prun <-
table(predict(tree_def_min_prun,type = "class"), train
_data_min_prun$annual_inc, dnn = c("predicted", "actual"))
pred_train_split <- table(predict(tree_def_split,type = "class"), train_data_
min_prun$annual_inc, dnn = c("predicted", "actual")) pred_test_min_prun <-
table(predict(tree_def_min_prun,test_data_min_prun,type = "class"),
test_data_min_prun$annual_inc, dnn = c("predicted", "actual"))
pred_test_split <- table(predict(tree_def_split,test_data_min_prun,type = "cl
ass"), test_data_min_prun$annual_inc, dnn = c("predicted", "actual"))

e1 <- 1-sum(diag(pred_train_min_prun))/sum(pred_train_min_prun)
e2 <- 1-sum(diag(pred_train_split))/sum(pred_train_split) e3 <-
1-sum(diag(pred_test_min_prun))/sum(pred_test_min_prun) e4 <-
1-sum(diag(pred_test_split))/sum(pred_test_split)

error_matrix_both <- matrix(c(e1,e2,e3,e4),ncol = 2,byrow = T)
colnames(error_matrix_both) <- c("Minimum pruned model","second model")
row.names(error_matrix_both) <- c("train","test")


#error results of both models
error_matrix_both
```

```
##       Minimum pruned model second model
## train             0.603512    0.6914972
## test              0.667597    0.6943902
```

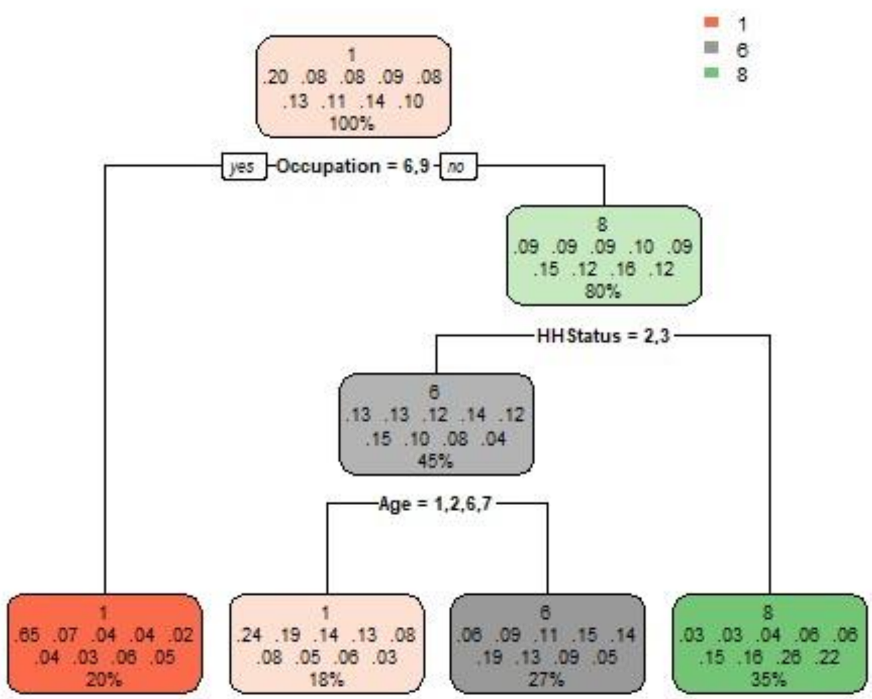
```r
# There is a difference in training errors of both models.
# Also, in the second model, training and test errors are almost same,
# but that is not tha case in minimum pruned model
# Although test error and training error is low for the minimum pruned model,
# there might be overfitting scenario which is reflected by the difference in
training and
# test errors and also the size of terminal nodesin the minimum pruned model
is much higher(52)
# compared to 4 terminal nodes in other model
```

```
### PART-K ##############################

set.seed(1234)

# 50-50 split
ind = sample(2, nrow(inc_data), replace = TRUE, prob = c(0.5,0.5))
train_data_50 = inc_data[ind == 1,] test_data_50 = inc_data[ind ==
2,]
tree_def_50 = rpart(annual_inc~., data = train_data_50 ,method = "class")

# training error and testing error
pred_train_50 <- table(predict(tree_def_50,type = "class"), train_data_50$ann
ual_inc, dnn = c("predicted", "actual"))
pred_test_50 <- table(predict(tree_def_50,test_data_50,type = "class"), test_
data_50$annual_inc, dnn = c("predicted", "actual")) err_train_50 <- 1-
sum(diag(pred_train_50))/sum(pred_train_50) err_test_50 <- 1-
sum(diag(pred_test_50))/sum(pred_test_50)

# 70-30 split
ind = sample(2, nrow(inc_data), replace = TRUE, prob = c(0.7,0.3))
train_data_70 = inc_data[ind == 1,] test_data_70 = inc_data[ind ==
2,]
tree_def_70 = rpart(annual_inc~., data = train_data_70 ,method = "class")

# training error and testing error
pred_train_70 <- table(predict(tree_def_70,type = "class"), train_data_70$ann
ual_inc, dnn = c("predicted", "actual"))
pred_test_70 <- table(predict(tree_def_70,test_data_70,type = "class"), test_
data_70$annual_inc, dnn = c("predicted", "actual")) err_train_70 <- 1-
sum(diag(pred_train_70))/sum(pred_train_70) err_test_70 <- 1-
sum(diag(pred_test_70))/sum(pred_test_70)

# 90-10 split
ind = sample(2, nrow(inc_data), replace = TRUE, prob = c(0.9,0.1))
train_data_90 = inc_data[ind == 1,] test_data_90 = inc_data[ind ==
2,]
tree_def_90 = rpart(annual_inc~., data = train_data_90 ,method = "class")

# training error and testing error
pred_train_90 <- table(predict(tree_def_90,type = "class"), train_data_90$ann
ual_inc, dnn = c("predicted", "actual"))
pred_test_90 <- table(predict(tree_def_90,test_data_90,type = "class"), test_
data_90$annual_inc, dnn = c("predicted", "actual")) err_train_90 <- 1-
sum(diag(pred_train_90))/sum(pred_train_90) err_test_90 <- 1-
sum(diag(pred_test_90))/sum(pred_test_90)
```

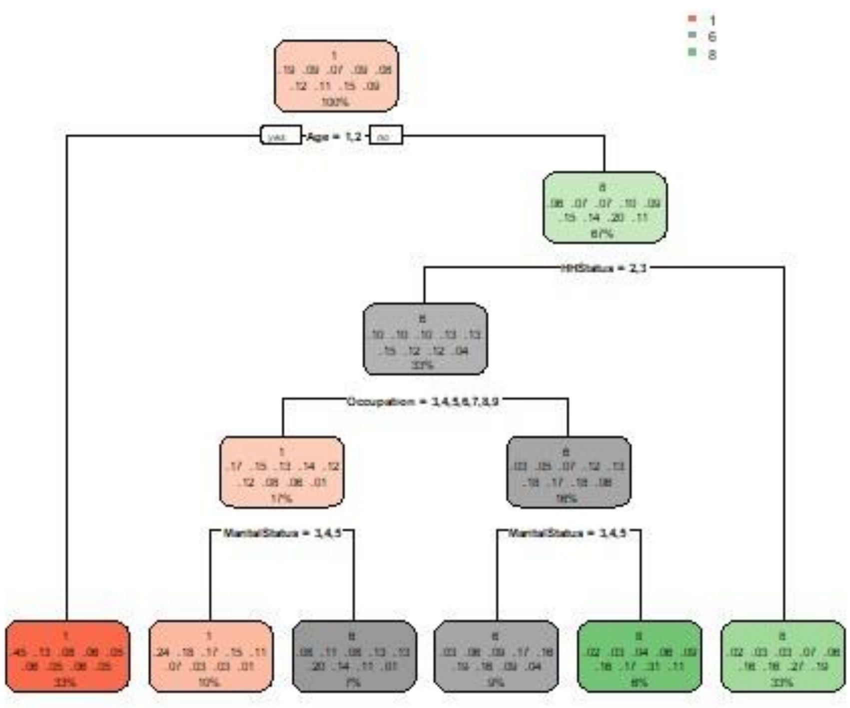
```
error_matrix_all_splits <- matrix(c(err_train_50,err_test_50,
                                    err_train_70,err_test_70,err_train_90,err
_test_90),ncol=2,byrow = T)
row.names(error_matrix_all_splits) <- c("split_50","split_70","split_90")
colnames(error_matrix_all_splits) <- c("train","test")
error_matrix_all_splits
```

```
##              train      test
## split_50 0.6844671 0.6979911
## split_70 0.6867991 0.7048346
## split_90 0.6906023 0.7082380
```

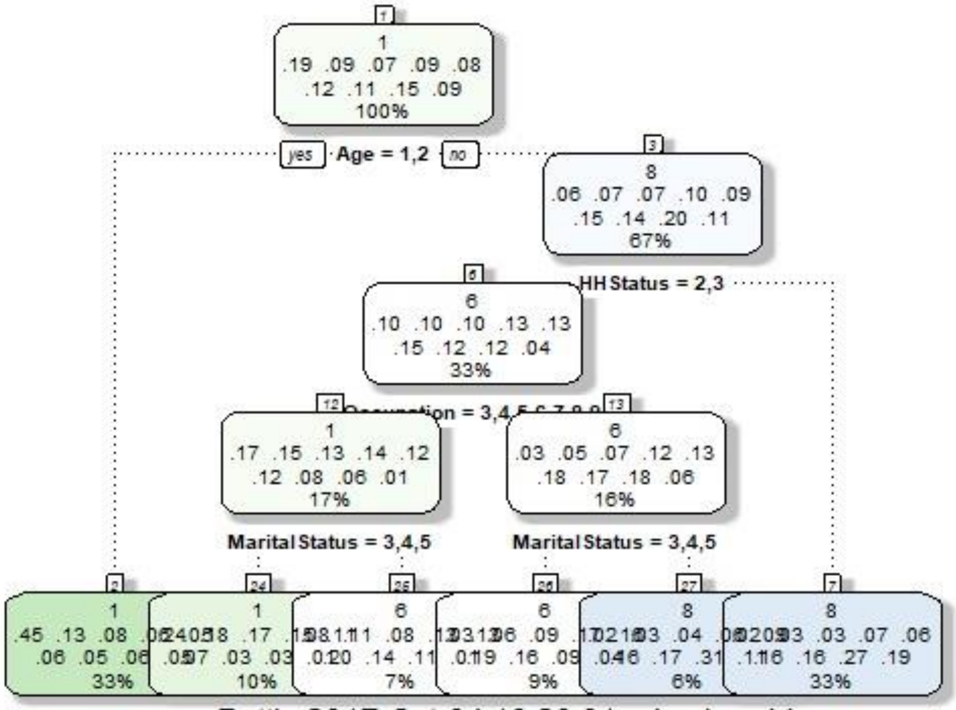
```
# error on training and testing sets seem to increase as we move from 50 to 9
0,
# I would prefer splitting at 50% beacause the training and test errors are c
omparatively
# low and number of observations in the terminal nodes have atleast 10%  of t
otal,
# which is acceptable also rpart.plot(tree_def_50)
```



```
### PART-L ##############################

# For Information gain using 60-40 split
tree_info_gain = rpart(annual_inc~., data = TrainData,method = "class", parms
= list(split = 'information'))
rpart.plot(tree_info_gain)
```




```
fancyRpartPlot(tree_info_gain,tweak=1.5)
```



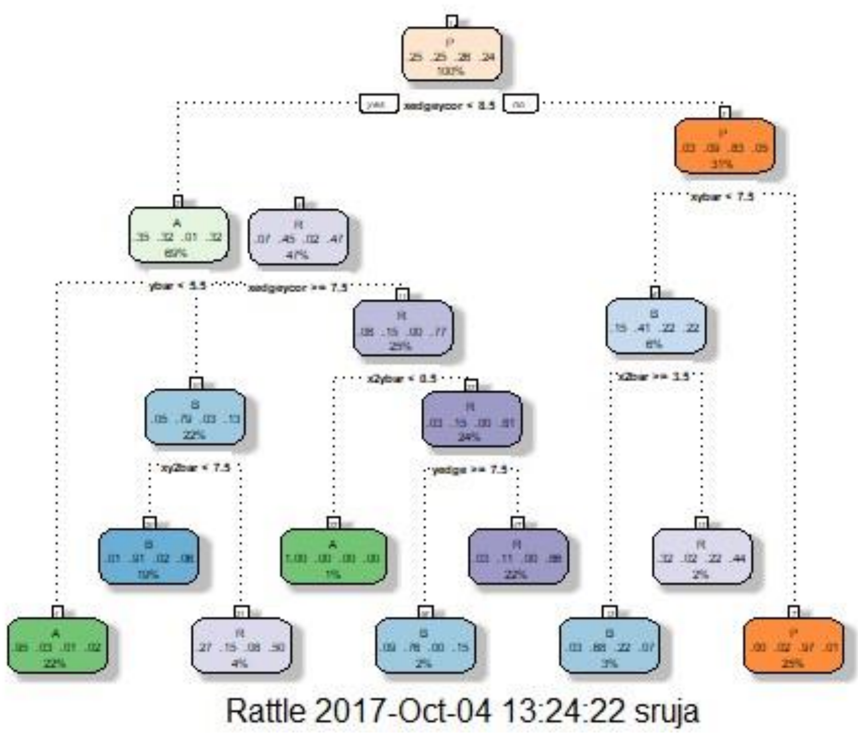

Rattle 2017-Oct-04 13:20:01 rakesh reddy

```
pred_train_info_gain <- table(predict(tree_info_gain,type = "class"), TrainDa
ta$annual_inc, dnn = c("predicted", "actual"))
pred_test_info_gain <- table(predict(tree_info_gain,TestData,type = "class"),
TestData$annual_inc, dnn = c("predicted", "actual"))

error_train_info <- 1-sum(diag(pred_train_info_gain))/sum(pred_train_info_gai
n)                    error_test_info                   <-                   1-
sum(diag(pred_test_info_gain))/sum(pred_test_info_gain)

error_train_info  ##

[1] 0.683549

error_test_info

## [1] 0.6837845

# default gini tree
pred_def_train_gini <- table(predict(tree_default,type = "class"), TrainData$
annual_inc, dnn = c("predicted", "actual"))
pred_def_test_gini <- table(predict(tree_default,TestData,type = "class"), Te
stData$annual_inc, dnn = c("predicted", "actual")) error_train_gini <- 1-
sum(diag(pred_def_train_gini))/sum(pred_def_train_gini) error_test_gini <- 1-
sum(diag(pred_def_test_gini))/sum(pred_def_test_gini)

error_matrix_info_gini <- matrix(c(error_train_info,error_test_info,
error_train_gini,error_test_gini),ncol=2,b yrow = T)

row.names(error_matrix_info_gini) <- c("info_Gain","gini")
colnames(error_matrix_info_gini) <- c("train","test")



error_matrix_info_gini



##                 train      test
## info_Gain 0.6835490 0.6837845
## gini      0.6914972 0.6943902



# Information gain shows a slight improvement in terms of error percentages c

ompared to gini
```

# assignment_2_q2

```
#We shall start by reading the dataset and construction using rpart.
letters_ABPR = read.csv('C:/Users/sruja/Downloads/letters_ABPR.csv')
set.seed(1234)

## 70% of the sample size to classify Training Data and Test Data smp_size
<- floor(0.70 * nrow(letters_ABPR))

## set the seed to make your partition reproductible train_ind <-
sample(seq_len(nrow(letters_ABPR)), size = smp_size)
TrainData <- letters_ABPR[train_ind, ]
TestData <- letters_ABPR[-train_ind, ]

library(rpart)
letters_ABPR$letter = as.factor(letters_ABPR$letter)

mytree <- rpart(letter ~.-letter, data = TrainData, method="class")

library(rattle)

## Warning: package 'rattle' was built under R version 3.4.2

## Rattle: A free graphical interface for data science with R.
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.4.2

library(RColorBrewer) fancyRpartPlot(mytree)
predicted_test_data = predict(mytree,newdata = TestData,type="class")
accuracy_r_part = table(TestData$letter,predicted_test_data)



## Decision Tree using C Tree

# Same dataset shall now be used to construct a decision tree using ctree.
library(party)

## Warning: package 'party' was built under R version 3.4.2

## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4
```

```
## Loading required package: strucchange

## Warning: package 'strucchange' was built under R version 3.4.2

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.4.2

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

## Warning: package 'sandwich' was built under R version 3.4.2
```




```
set.seed(1000) letters_ctree  = ctree(letter ~.-letter,
data=TrainData) c_tree_test_data = predict(letters_ctree,
newdata=TestData) accuracy_c_tree =
table(TestData$letter,c_tree_test_data)


## Random Forest
# Now we can use the same data and construct the random forest for the same.
```

```r
library("randomForest")

## Warning: package 'randomForest' was built under R version 3.4.2

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:rattle':
##
##     importance

set.seed(1000) rfModel = randomForest(letter ~.-letter,
data=TrainData) rfModelTestData =
predict(rfModel,newdata=TestData)
accuracy_random_forest = table(TestData$letter,rfModelTestData)

# Variable Importance
# Importance of Variables in Random Forest

rfModel$importance


##             MeanDecreaseGini
##  xbox                 18.918
19.05121 ## width
## height            20.06740
## onpix             20.20779
## xbar              33.84695
## ybar             251.27470
## x2bar             58.11724
## y2bar            110.55706    80.38886
## x2ybar           171.95234
## xy2bar           222.07456
## xedge             74.10662
## xedgeycor        322.30306
## yedge            130.03108
## yedgexcor         86.84553

#Variable Impor

summary(mytree)
```

```
## Call:                              ce from summary of rpart
## rpart(formula = letter ~
##   n= 2181
##
                                      - letter, data = TrainData, method = "class")


                                      plit rel error    xerror
                                       xstd
                                         0 1.0000000 1.0000000
                                      012797354
```

```
## 2 0.25762290      1 0.6869944 0.6869944 0.014529433
## 3 0.20224020      2 0.4293715 0.4293715 0.013515112
## 4 0.01680149      3 0.2271313 0.2271313 0.010848274
## 5 0.01555694      4 0.2103298 0.2190417 0.010691397
## 6 0.01431238      5 0.1947729 0.2084630 0.010478390
## 7 0.01306783      6 0.1804605 0.1860610 0.009995435 ##
8 0.01000000      8 0.1543248 0.1636590 0.009463646 ##

## Variable importance

##       ybar xedgeycor     x2ybar     xy2bar      yedge      y2bar      xedge
##         18        17         14         12         10          8          7

##      xybar      x2bar       xbar yedgexcor
##          5          5          3          1
```

```
##
## Node number 1: 2181 observations,    complexi       (node) =1
##   predicted class=P  expected loss=0.736818
##     class counts:   542   541   574   524
##    probabilities: 0.249 0.248 0.263 0.240 ##
son=2 (1513 obs) right son=3 (668 obs) ##    P
splits:
##       xedgeycor < 8.5  to the left,  improve=ıt, improve=396.8407, (0
##       xy2bar    <                                            missing)
##       ybar      < 5.5  to the left,  improve=
##        x2ybar    < 2.5  to the left,  improv
yedge     < 4.5  to the left,  improve=231.2473,                missing)
##       xy2bar < 5.5  to the right, agree=0.896
##       ybar   < 8.5  to the left,  agree=0.834
##        xedge  < 1.5  to the right, agree=0.8
x2ybar < 6.5  to the left,  agree=0.783, adj=0.2
< 10.5 to the left,  agree=0.776, adj=0.268, (0
## Node number 2: 1513 observations,    complexi
##   predicted class=A  expected loss=0.6543291
##     class counts:   523   479    20   491
##    probabilities: 0.346 0.317 0.013 0.325 ##
son=4 (481 obs) right son=5 (1032 obs) ##    P             e) =0.6937185
splits:
##        ybar       < 5.5  to the left,  improv
x2ybar    < 2.5  to the left,  improve=322.3396,
##       y2bar     < 3.5  to the left,  improve=
##       yedge     < 3.5  to the left,  improve=
##        xedgeycor < 7.5  to the right, improv
Surrogate splits:
##       x2ybar < 2.5  to the left,  agree=0.904
##       y2bar  < 2.5  to the left,  agree=0.890                (0 missing)
##       yedge  < 3.5  to the left,  agree=0.868
##        x2bar  < 2.5  to the left,  agree=0.8
xbar   < 9.5  to the right, agree=0.776, adj=0.2
##
```

```
## Node number 3: 668 observations,    complexity param=0.01431238
##   predicted class=P  expected loss=0.1706587  P(node) =0.3062815
##     class counts:    19    62   554    33
##    probabilities: 0.028 0.093 0.829 0.049  ##
left son=6 (122 obs) right son=7 (546 obs) ##
Primary splits:
##       xybar  < 7.5  to the left,  improve=76.71080, (0 missing)
##       xy2bar < 6.5  to the right, improve=71.00796, (0 missing)
##       yedge  < 6.5  to the right, improve=66.17969, (0 missing)
##       ybar   < 7.5  to the left,  improve=59.38806, (0 missing) ##
xedge  < 5.5  to the right, improve=42.32027, (0 missing) ##
Surrogate splits:
##       xy2bar    < 6.5  to the right, agree=0.930, adj=0.615, (0 split)
##       xedge     < 5.5  to the right, agree=0.897, adj=0.434, (0 split)
##       yedge     < 6.5  to the right, agree=0.888, adj=0.385, (0 split)
##       ybar      < 7.5  to the left,  agree=0.885, adj=0.369, (0 split) ##
yedgexcor < 5.5  to the left,  agree=0.850, adj=0.180, (0 split) ##
## Node number 4: 481 observations
##   predicted class=A  expected loss=0.05405405  P(node) =0.220541
##     class counts:   455    13     4     9
##    probabilities: 0.946 0.027 0.008 0.019  ##
## Node number 5: 1032 observations,    complexity param=0.2022402
##   predicted class=R  expected loss=0.5329457  P(node) =0.4731774
##     class counts:    68   466    16   482
##    probabilities: 0.066 0.452 0.016 0.467  ##
left son=10 (490 obs) right son=11 (542 obs) ##
Primary splits:
##       xedgeycor < 7.5  to the right, improve=215.41000, (0 missing)
##       x2ybar    < 5.5  to the right, improve=115.13940, (0 missing)
##       xy2bar    < 7.5  to the left,  improve= 92.49908, (0 missing)
##       xedge     < 2.5  to the left,  improve= 69.48348, (0 missing) ##
yedge     < 6.5  to the right, improve= 68.95931, (0 missing) ##
Surrogate splits:
##       x2ybar < 5.5  to the right, agree=0.734, adj=0.441, (0 split)
##       xedge  < 2.5  to the left,  agree=0.695, adj=0.357, (0 split)
##       xy2bar < 7.5  to the left,  agree=0.644, adj=0.251, (0 split)
##       yedge  < 5.5  to the right, agree=0.641, adj=0.243, (0 split) ##
ybar   < 7.5  to the left,  agree=0.622, adj=0.204, (0 split) ##
## Node number 6: 122 observations,    complexity param=0.01306783
##   predicted class=B  expected loss=0.5901639  P(node) =0.05593764
##     class counts:    18    50    27    27
##    probabilities: 0.148 0.410 0.221 0.221  ##
left son=12 (72 obs) right son=13 (50 obs) ##
Primary splits:
##       x2bar  < 3.5  to the right, improve=19.44719, (0 missing)
##       yedge  < 4.5  to the left,  improve=16.97641, (0 missing)
##       height < 8.5  to the left,  improve=11.24709, (0 missing)
```

```
##       xy2bar < 7.5  to the left,  improve=11.15211, (0 missing) ##
x2ybar < 7.5  to the right, improve=10.44491, (0 missing) ##
Surrogate splits:
##       x2ybar    < 7.5  to the left,  agree=0.738, adj=0.36, (0 split)
##       yedgexcor < 5.5  to the right, agree=0.738, adj=0.36, (0 split)
##       xy2bar    < 8.5  to the left,  agree=0.713, adj=0.30, (0 split)
##       yedge     < 5.5  to the right, agree=0.713, adj=0.30, (0 split) ##
ybar      < 7.5  to the left,  agree=0.631, adj=0.10, (0 split) ##
## Node number 7: 546 observations
##   predicted class=P  expected loss=0.03479853  P(node) =0.2503439
##     class counts:     1    12   527     6
##    probabilities: 0.002 0.022 0.965 0.011  ##
## Node number 10: 490 observations,    complexity param=0.01680149
##   predicted class=B  expected loss=0.2102041  P(node) =0.2246676
##     class counts:    25   387    16    62
##    probabilities: 0.051 0.790 0.033 0.127  ##
left son=20 (412 obs) right son=21 (78 obs) ##
Primary splits:
##       xy2bar    < 7.5  to the left,  improve=55.05601, (0 missing)
##       xedge     < 2.5  to the left,  improve=41.89844, (0 missing)
##       y2bar     < 4.5  to the right, improve=33.63993, (0 missing)
##       yedgexcor < 5.5  to the left,  improve=26.96594, (0 missing) ##
ybar      < 8.5  to the left,  improve=18.60346, (0 missing) ##
Surrogate splits:
##       yedgexcor < 4.5  to the right, agree=0.871, adj=0.192, (0 split)
##       ybar      < 9.5  to the left,  agree=0.861, adj=0.128, (0 split)
##       xedge     < 5.5  to the left,  agree=0.859, adj=0.115, (0 split)
##       ybox      < 11.5 to the left,  agree=0.849, adj=0.051, (0 split) ##
xbar      < 5.5  to the right, agree=0.849, adj=0.051, (0 split) ##
## Node number 11: 542 observations,    complexity param=0.01555694
##   predicted class=R  expected loss=0.2250923  P(node) =0.2485099
##     class counts:    43    79     0   420
##    probabilities: 0.079 0.146 0.000 0.775  ##
left son=22 (25 obs) right son=23 (517 obs) ##
Primary splits:
##       x2ybar    < 0.5  to the left,  improve=38.51003, (0 missing)
##       y2bar     < 1.5  to the left,  improve=36.64683, (0 missing)
##       yedge     < 2.5  to the left,  improve=34.92632, (0 missing)
##       yedgexcor < 8.5  to the left,  improve=19.19285, (0 missing) ##
ybar      < 6.5  to the left,  improve=18.23345, (0 missing) ##
Surrogate splits:
##       y2bar < 1.5  to the left,  agree=0.998, adj=0.96, (0 split) ##
yedge < 2.5  to the left,  agree=0.996, adj=0.92, (0 split) ##
## Node number 12: 72 observations
##   predicted class=B  expected loss=0.3194444  P(node) =0.03301238 ##
class counts:     2    49    16     5
```

```
##    probabilities: 0.028 0.681 0.222 0.069  ##
## Node number 13: 50 observations
##   predicted class=R  expected loss=0.56  P(node) =0.02292526
##     class counts:    16     1    11    22
##    probabilities: 0.320 0.020 0.220 0.440  ##
## Node number 20: 412 observations
##   predicted class=B  expected loss=0.08980583  P(node) =0.1889042
##     class counts:     4   375    10    23
##    probabilities: 0.010 0.910 0.024 0.056  ##
## Node number 21: 78 observations
##   predicted class=R  expected loss=0.5  P(node) =0.03576341
##     class counts:    21    12     6    39
##    probabilities: 0.269 0.154 0.077 0.500  ##
## Node number 22: 25 observations
##   predicted class=A  expected loss=0  P(node) =0.01146263
##     class counts:    25     0     0     0
##    probabilities: 1.000 0.000 0.000 0.000  ##
## Node number 23: 517 observations,    complexity param=0.01306783 ##
predicted class=R  expected loss=0.1876209  P(node) =0.2370472
##     class counts:    18    79     0   420
##    probabilities: 0.035 0.153 0.000 0.812  ##
left son=46 (34 obs) right son=47 (483 obs) ##
Primary splits:
##       yedge  < 7.5  to the right, improve=29.83994, (0 missing)
##       x2ybar < 5.5  to the right, improve=21.51856, (0 missing)
##       xedge  < 3.5  to the right, improve=21.35032, (0 missing)
##       xbox   < 4.5  to the right, improve=19.41873, (0 missing) ##
xybar  < 8.5  to the right, improve=16.45167, (0 missing) ##
Surrogate splits:
##       ybox      < 14.5 to the right, agree=0.940, adj=0.088, (0 split)
##       xedgeycor < 4.5  to the left,  agree=0.940, adj=0.088, (0 split) ##
xy2bar    < 5.5  to the left,  agree=0.938, adj=0.059, (0 split) ##
## Node number 46: 34 observations
##   predicted class=B  expected loss=0.2352941  P(node) =0.01558918
##     class counts:     3    26     0     5
##    probabilities: 0.088 0.765 0.000 0.147  ##
## Node number 47: 483 observations
##   predicted class=R  expected loss=0.1407867  P(node) =0.221458
##     class counts:    15    53     0   415
##    probabilities: 0.031 0.110 0.000 0.859
```

```
# Comparing the important variables from using the above 2 commands we find t
hat the important variables are almost similiar. For random forest we conside
r the variables with highest mean decrease as important varaibles which would
yield us the following results. xedgeycor, ybar, xy2bar, x2ybar,yedge,y2bar,x
edge are considered the most important variables in both the cases. The prima
ry splits in rpart or based on  yedge, x2ybar,xedge,xbox,xybar and surrogate
splits are based on ybox,xedgeycor,xy2bar.

## Comparing Accuracies
# Now that we are done plotting the various decision trees and random forest
for the given dataset we have to evaluate which one would be giving us the ac
curate predictions. So we should find out the accuracy for each type of tree.

accuracy_c_tree

##    c_tree_test_data
##       A   B   P   R
##   A 247   0   0   0
##   B   0 225   0   0
##   P   0   0 229   0 ##
R   0   0   0 234

accuracy_r_part

##    predicted_test_data ##
A   B   P   R
##   A 220   3   1  23
##   B   4 183   5  33
##   P   1  16 205   7 ##
R   2  11   2 219

accuracy_random_forest

##    rfModelTestData
##       A   B   P   R
##   A 247   0   0   0
##   B   0 221   0   4
##   P   0   1 228   0 ##
R   0   3   0 231

error_accuracy_c_tree = sum(diag(accuracy_c_tree))/sum(accuracy_c_tree)*100
error_accuracy_r_part = sum(diag(accuracy_r_part))/sum(accuracy_r_part)*100
error_accuracy_random_forest = sum(diag(accuracy_random_forest))/sum(accuracy
_random_forest)*100

error_accuracy_c_tree
```

## [1] 100

error_accuracy_r_part ##

[1] 88.4492

error_accuracy_random_forest

```
## [1] 99.14439

# Calculating the accuracies for all the 3 models we find that the best accur
acy is for decision tree using c tree. It is giving a perfect accuracy with n
o errors. While Random forest is giving us a close to perfect model the decis
ion tree using rpart is close to 90%. We can  say that the best decision tree

for given dataset can be achieved through decision tree using ctree.


# We can say that the best model for this dataset is decision tree using c tr
ee which gives us a perfect model with no errors on Test data as well as trai

ning data.
```