R - RF/DT K-Fold and Evaluation Charts

Karthick Sharan

3/17/2022

Importing libraries

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(DataExplorer)
```

```
## Warning: package 'DataExplorer' was built under R version 4.1.3
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(ggcorrplot)
library(mice)
```

```
## Warning: package 'mice' was built under R version 4.1.3

##
## Attaching package: 'mice'

## The following object is masked from 'package:stats':
##
##     filter

## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.3

## randomForest 4.7-1

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

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
library(rpart)
library(rpart.plot)
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.1.3
```

```r
library(caret)
```

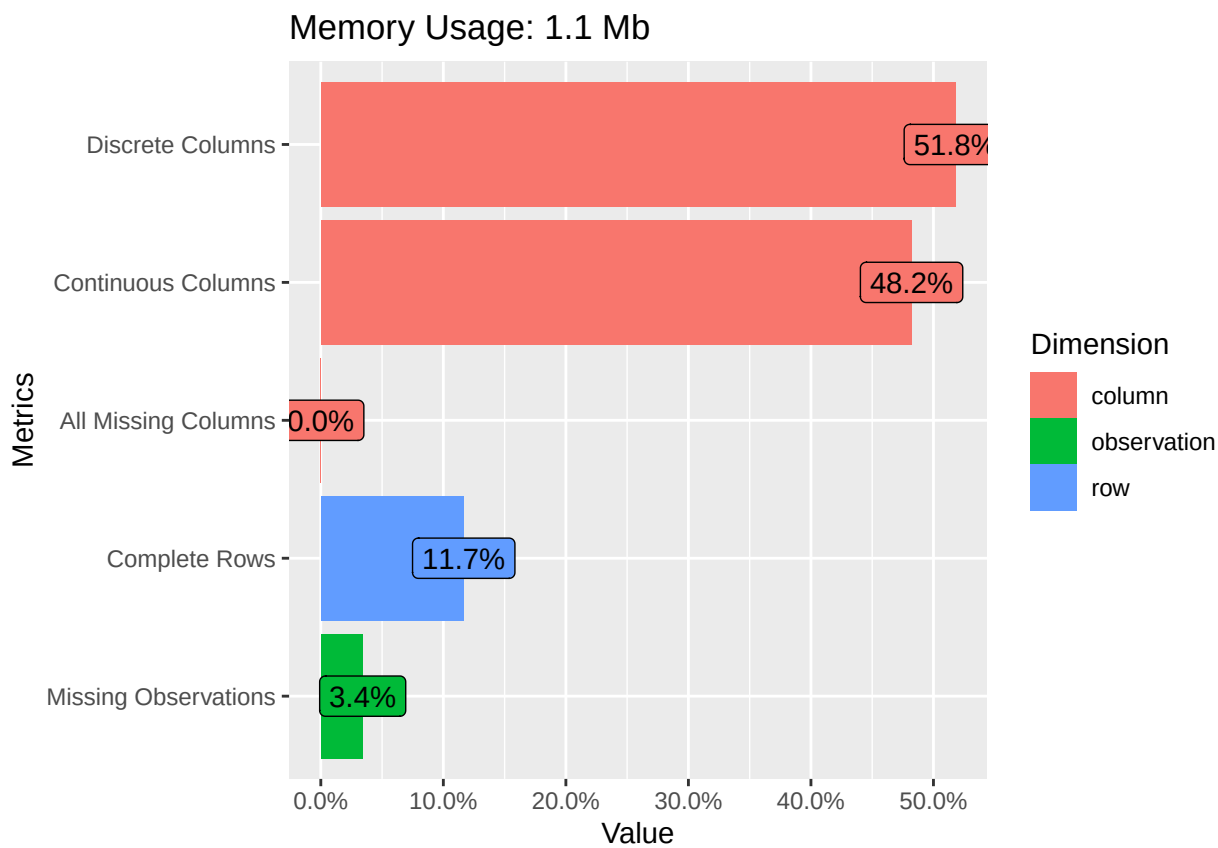
```
## Warning: package 'caret' was built under R version 4.1.3

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

DATA EXPLORATION

```r
# Reading dataset
df <- read.csv("C:/Users/rshara4/Desktop/hw3/dataset.csv")
col <- names(df)
introduce(df)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 2392      56               29                 27                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                 4577           279             133952      1104736
```

```r
plot_intro(df)
```




```
dev.new(width=5, height=8)
plot_missing(df)
```

```
drop_col <- c("ï..ID","Special.Pay","Initial.System.Date", "FirstMeeting","LastMeeting",
          "SchoolGradeTypeLow","SchoolGradeTypeHigh","SchoolGradeType","GroupGradeTypeLow",
          "GroupGradeTypeHigh","Num.of.Non_FPP.PAX","Total.Pax", "Departure.Date",
          "Return.Date","Group.State","GroupGradeType")
dfn <- select(df, -drop_col)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(drop_col)‘ instead of ‘drop_col‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
# Reasons for dropping above columns: Redundant columns, feature extraction,
# unique columns, high % of Missing values.
```

```
# Creating bucket in program code to reduce categories
table(dfn['Program.Code'])
```

```
##
##        CC   CD   CN  CVP   FN   HC   HD   HF   HG   HH  HIS   HN   HO   HP   HS
##     3  14  114    4    3    1  274 1430    6   25    3    5   73   11    3  131
##   HVP   HW   HX   HY   HZ   IC   IK   SC   SD   SG   SK   SM   ST
##    62    9    3    7    7   12    3   46    4   20   68   23   28
```

```r
code <- c("HD","HC","CD","HN","HS","HVP","SK","SC") #choosing top 8 categories
dfn$Program.Code <- ifelse(dfn$Program.Code %in% code, dfn$Program.Code,"Others")
#converting other subgroups to "Others"
table(dfn['Program.Code'])
```

```
##
##     CD     HC     HD     HN     HS    HVP Others     SC     SK
##    114    274   1430     73    131     62    194     46     68
```

```r
date_col <- c("Early.RPL","Latest.RPL")
dfn[,date_col] <- lapply(df[,date_col], function(x) as.Date(x, format="%m/%d/%Y"))
str(dfn)
```

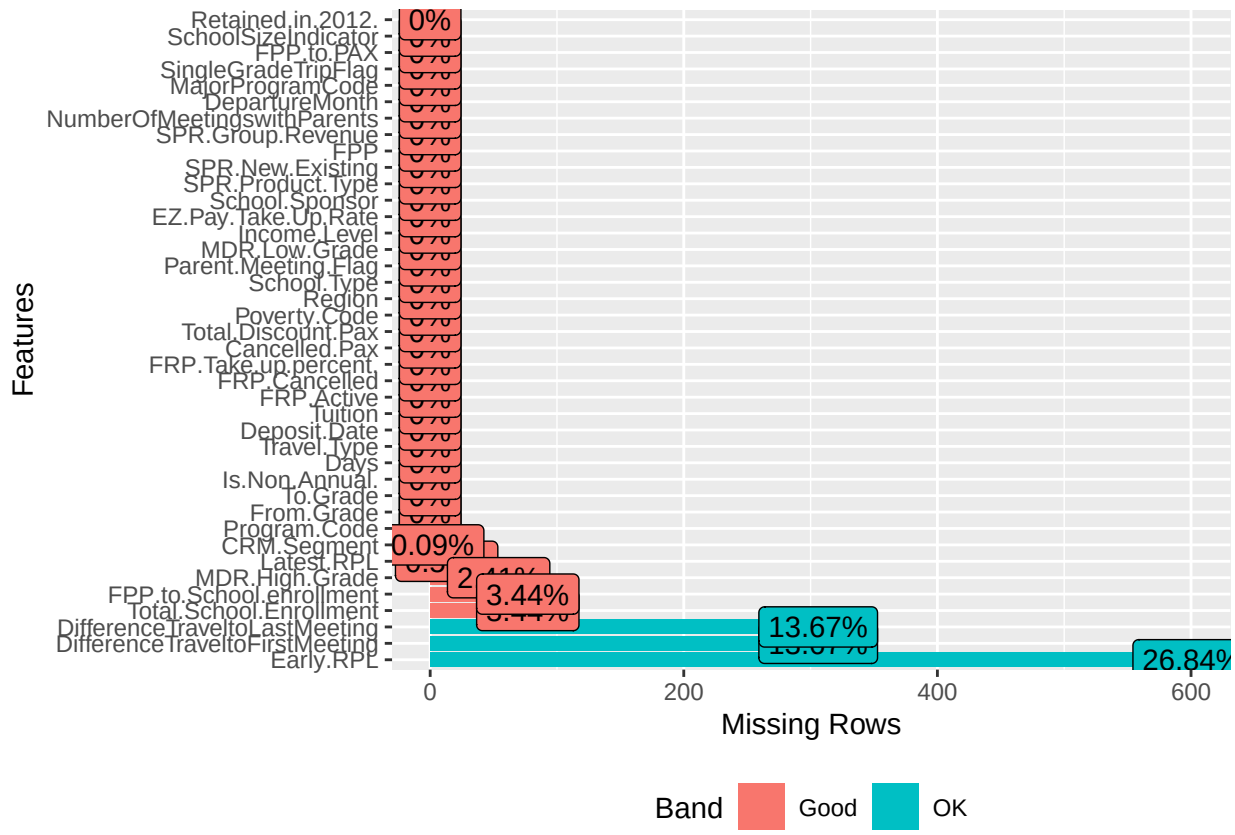
```
## 'data.frame':    2392 obs. of  40 variables:
##  $ Program.Code                  : chr  "HS" "HC" "HD" "HN" ...
##  $ From.Grade                    : int  4 8 8 9 6 10 11 9 8 8 ...
##  $ To.Grade                      : int  4 8 8 12 8 12 12 9 8 8 ...
##  $ Is.Non.Annual.                : int  0 0 0 1 0 0 1 0 0 0 ...
##  $ Days                          : int  1 7 3 3 6 4 6 8 8 4 ...
##  $ Travel.Type                   : chr  "A" "A" "A" "B" ...
##  $ Deposit.Date                  : chr  "8/30/2010" "11/15/2009" "10/15/2010" "1/7/2011" ...
##  $ Tuition                       : int  424 2350 1181 376 865 2025 1977 3379 2200 1428 ...
##  $ FRP.Active                    : int  25 9 17 0 40 9 16 10 30 51 ...
##  $ FRP.Cancelled                 : int  3 9 6 0 8 4 4 0 0 1 ...
##  $ FRP.Take.up.percent.          : num  0.424 0.409 0.708 0 0.494 0.9 0.64 0.769 0.577 0.773 ...
##  $ Early.RPL                     : Date, format: "2010-03-29" "2009-10-20" ...
##  $ Latest.RPL                    : Date, format: "2010-08-12" "2010-08-10" ...
##  $ Cancelled.Pax                 : int  3 11 6 1 9 3 5 1 0 1 ...
##  $ Total.Discount.Pax            : int  4 3 3 0 8 1 2 1 4 6 ...
##  $ Poverty.Code                  : chr  "B" "C" "C" "" ...
##  $ Region                        : chr  "Southern California" "Other" "Other" "Other" ...
##  $ CRM.Segment                   : int  4 10 10 7 10 8 8 7 5 5 ...
##  $ School.Type                   : chr  "PUBLIC" "PUBLIC" "PUBLIC" "CHD" ...
##  $ Parent.Meeting.Flag           : int  1 1 1 0 1 1 1 1 1 1 ...
##  $ MDR.Low.Grade                 : chr  "K" "7" "6" "" ...
##  $ MDR.High.Grade                : int  5 8 8 NA 8 12 12 NA 12 8 ...
##  $ Total.School.Enrollment       : int  927 850 955 NA 720 939 225 NA 500 635 ...
##  $ Income.Level                  : chr  "Q" "A" "O" "" ...
##  $ EZ.Pay.Take.Up.Rate           : num  0.17 0.091 0.042 0 0.383 0.1 0.08 0 0.231 0.136 ...
##  $ School.Sponsor                : int  1 0 0 0 0 0 0 0 0 0 ...
##  $ SPR.Product.Type              : chr  "CA History" "East Coast" "East Coast" "East Coast" ...
##  $ SPR.New.Existing              : chr  "EXISTING" "EXISTING" "EXISTING" "EXISTING" ...
##  $ FPP                           : int  59 22 24 18 81 10 25 13 52 66 ...
##  $ SPR.Group.Revenue             : int  424 2350 1181 376 865 2025 1977 3379 2200 1428 ...
##  $ NumberOfMeetingswithParents   : int  1 2 1 0 1 1 1 1 1 1 ...
##  $ DifferenceTraveltoFirstMeeting: int  155 423 124 NA 145 91 63 138 143 146 ...
##  $ DifferenceTraveltoLastMeeting : int  155 140 124 NA 145 91 63 138 143 146 ...
##  $ DepartureMonth                : chr  "January" "January" "January" "January" ...
##  $ MajorProgramCode              : chr  "H" "H" "H" "H" ...
##  $ SingleGradeTripFlag           : int  1 1 1 0 0 0 0 1 1 1 ...
##  $ FPP.to.School.enrollment      : num  0.0636 0.0259 0.0251 NA 0.1125 ...
##  $ FPP.to.PAX                    : num  0.937 0.88 0.889 1 0.91 ...
##  $ SchoolSizeIndicator           : chr  "L" "L" "L" "" ...
```

```
##  $ Retained.in.2012.             : int  1 1 1 0 0 1 0 0 1 1 ...
```

```
dfn <- dfn %>% drop_na(c("From.Grade","To.Grade","Deposit.Date")) #drop rows with NA
introduce(dfn)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 2239      40               15                 25                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                 1436          1441              89560       551752
```

```
plot_missing(dfn)
```




```
# FEATURE EXTRACTIONS :

range_elem <- c(1,2,3,4,5)
range_mid <- c(6,7,8,9)
range_high <- c(10,11,12)

dfn <- dfn %>%
  mutate(grade_low = case_when(
    From.Grade %in% range_elem ~ "E",
    From.Grade %in% range_mid ~ "M",
    From.Grade %in% range_high ~ "H"
  ))
```

```r
dfn <- dfn %>%
  mutate(grade_high = case_when(
    To.Grade %in% range_elem ~ "E",
    To.Grade %in% range_mid ~ "M",
    To.Grade %in% range_high ~ "H"
  ))


dfn <- dfn %>%
  mutate(MDR_low = case_when(
    MDR.Low.Grade %in% range_elem ~ "E",
    MDR.Low.Grade %in% range_mid ~ "M",
    MDR.Low.Grade %in% range_high ~ "H",
    TRUE ~ MDR.Low.Grade
  ))
dfn <- dfn %>%
  mutate(MDR_high = case_when(
    MDR.High.Grade %in% range_elem ~ "E",
    MDR.High.Grade %in% range_mid ~ "M",
    MDR.High.Grade %in% range_high ~ "H",
  ))

dfn$Deposit_month <- month(mdy(dfn$Deposit.Date))


dfn <- dfn %>%
  mutate(grade_range = paste(grade_low,"-",grade_high)) %>%
  mutate(FRP_Actual = FRP.Active - FRP.Cancelled) %>%
  mutate(RPL_diff = difftime(Latest.RPL,Early.RPL, units = "days")) %>%
  mutate(MDR_range = paste(MDR_low,"-",MDR_high))

# 4 new columns created from the 8, so in total we'll reduce 4 columns this way,
# while capturing the relevant information
```

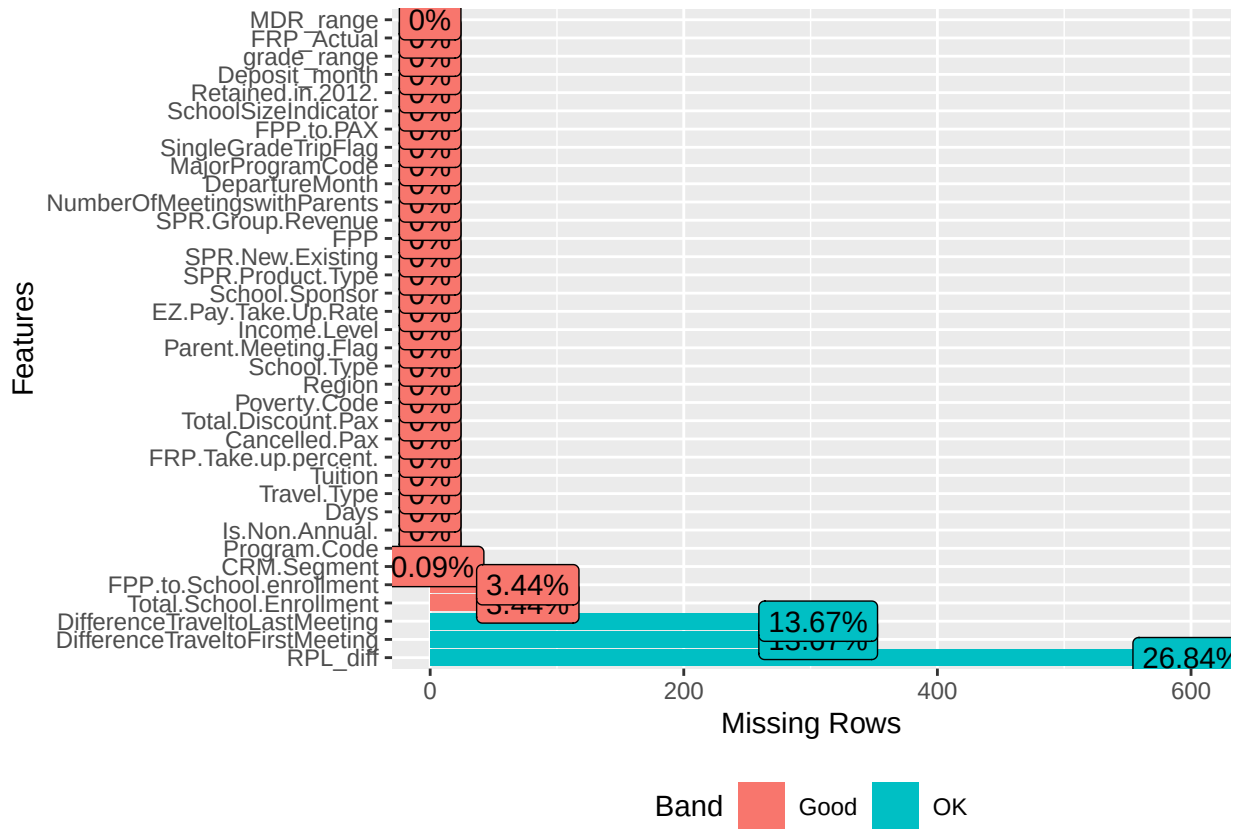
```r
drop_after_extraction <- c("Deposit.Date","From.Grade","To.Grade","FRP.Active","FRP.Cancelled",
                           "Early.RPL","Latest.RPL","MDR.Low.Grade","MDR.High.Grade",
                           "grade_low","grade_high","MDR_low","MDR_high")
dfn <- select(dfn, -drop_after_extraction)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(drop_after_extraction)` instead of `drop_after_extraction` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```r
introduce(dfn)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 2239      36               14                 22                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                 1369          1442              80604       507200
```

```
plot_missing(dfn)
```




```
col <- names(dfn)
cat <- names(dfn[, sapply(dfn, class) == 'character'])

# some categorical columns are missclassified as numerical, hence converting
conv <- c("Is.Non.Annual.","Parent.Meeting.Flag","School.Sponsor",
          "SingleGradeTripFlag" ,"Retained.in.2012.","Deposit_month",
          "NumberOfMeetingswithParents") #other category are ordinal

cat <- c(cat,conv)
num <- names(select(dfn, -cat)) #numerical columns
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(cat)‘ instead of ‘cat‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
dfn[cat] <- lapply(dfn[cat], factor)

str(dfn)
```

```
## ’data.frame’:    2239 obs. of  36 variables:
##  $ Program.Code                  : Factor w/ 9 levels "CD","HC","HD",..: 5 2 3 4 3 2 7 7 7 3 ...
```

```
##  $ Is.Non.Annual.                : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 2 1 1 1 ...
##  $ Days                          : int  1 7 3 3 6 4 6 8 8 4 ...
##  $ Travel.Type                   : Factor w/ 4 levels "A","B","N","T": 1 1 1 2 4 1 1 1 1 1 ...
##  $ Tuition                       : int  424 2350 1181 376 865 2025 1977 3379 2200 1428 ...
##  $ FRP.Take.up.percent.          : num  0.424 0.409 0.708 0 0.494 0.9 0.64 0.769 0.577 0.773 ...
##  $ Cancelled.Pax                 : int  3 11 6 1 9 3 5 1 0 1 ...
##  $ Total.Discount.Pax            : int  4 3 3 0 8 1 2 1 4 6 ...
##  $ Poverty.Code                  : Factor w/ 7 levels "","0","A","B",..: 4 5 5 1 6 5 1 1 1 1 ...
##  $ Region                        : Factor w/ 6 levels "Dallas","Houston",..: 6 4 4 4 4 4 4 4 4 2 ...
##  $ CRM.Segment                   : int  4 10 10 7 10 8 8 7 5 5 ...
##  $ School.Type                   : Factor w/ 4 levels "Catholic","CHD",..: 4 4 4 2 4 4 1 2 2 3 ...
##  $ Parent.Meeting.Flag           : Factor w/ 2 levels "0","1": 2 2 2 1 2 2 2 2 2 2 ...
##  $ Total.School.Enrollment       : int  927 850 955 NA 720 939 225 NA 500 635 ...
##  $ Income.Level                  : Factor w/ 23 levels "","A","B","C",..: 22 2 16 1 4 10 8 1 12 12 .
##  $ EZ.Pay.Take.Up.Rate           : num  0.17 0.091 0.042 0 0.383 0.1 0.08 0 0.231 0.136 ...
##  $ School.Sponsor                : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ SPR.Product.Type              : Factor w/ 6 levels "CA History","Costa Rica",..: 1 3 3 3 3 3 6 3 3
##  $ SPR.New.Existing              : Factor w/ 2 levels "EXISTING","NEW": 1 1 1 1 1 2 1 1 1 1 ...
##  $ FPP                           : int  59 22 24 18 81 10 25 13 52 66 ...
##  $ SPR.Group.Revenue             : int  424 2350 1181 376 865 2025 1977 3379 2200 1428 ...
##  $ NumberOfMeetingswithParents   : Factor w/ 3 levels "0","1","2": 2 3 2 1 2 2 2 2 2 2 ...
##  $ DifferenceTraveltoFirstMeeting: int  155 423 124 NA 145 91 63 138 143 146 ...
##  $ DifferenceTraveltoLastMeeting : int  155 140 124 NA 145 91 63 138 143 146 ...
##  $ DepartureMonth                : Factor w/ 6 levels "April","February",..: 3 3 3 3 3 3 3 3 3 2 ...
##  $ MajorProgramCode              : Factor w/ 4 levels "C","H","I","S": 2 2 2 2 2 2 4 3 1 2 ...
##  $ SingleGradeTripFlag           : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 1 2 2 2 ...
##  $ FPP.to.School.enrollment      : num  0.0636 0.0259 0.0251 NA 0.1125 ...
##  $ FPP.to.PAX                    : num  0.937 0.88 0.889 1 0.91 ...
##  $ SchoolSizeIndicator           : Factor w/ 5 levels "","L","M-L","S",..: 2 2 2 1 3 2 4 1 5 3 ...
##  $ Retained.in.2012.             : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 1 1 2 2 ...
##  $ Deposit_month                 : Factor w/ 12 levels "1","2","3","4",..: 8 11 10 1 9 9 10 10 6 10
##  $ grade_range                   : Factor w/ 6 levels "E - E","E - H",..: 1 6 6 5 6 4 4 6 6 6 ...
##  $ FRP_Actual                    : int  22 0 11 0 32 5 12 10 30 50 ...
##  $ RPL_diff                      : 'difftime' num  136 294 109 NA ...
##   ..- attr(*, "units")= chr "days"
##  $ MDR_range                     : Factor w/ 13 levels " - NA","E - E",..: 6 10 10 1 10 5 9 1 9 13 .
```

```
summary(dfn)
```

```
##   Program.Code  Is.Non.Annual.      Days        Travel.Type    Tuition
##  HD     :1354   0:1884         Min.   : 1.000   A:1908      Min.   :  79
##  HC     : 259   1: 355         1st Qu.: 4.000   B: 323      1st Qu.:1203
##  Others : 165                  Median : 5.000   N:   2      Median :1705
##  HS     : 129                  Mean   : 4.555   T:   6      Mean   :1618
##  CD     : 108                  3rd Qu.: 5.000               3rd Qu.:2049
##  HN     :  66                  Max.   :12.000               Max.   :4200
##  (Other): 158
##  FRP.Take.up.percent. Cancelled.Pax    Total.Discount.Pax Poverty.Code
##  Min.   :0.0000       Min.   : 0.000   Min.   : 0.000      :559
##  1st Qu.:0.4620       1st Qu.: 2.000   1st Qu.: 1.000     0:  4
##  Median :0.6000       Median : 4.000   Median : 2.000     A:238
##  Mean   :0.5755       Mean   : 4.808   Mean   : 2.935     B:919
##  3rd Qu.:0.7270       3rd Qu.: 6.000   3rd Qu.: 4.000     C:472
##  Max.   :1.0000       Max.   :39.000   Max.   :47.000     D: 33
```

```
##                                                          E: 14
##                  Region      CRM.Segment                    School.Type
##  Dallas             : 158   Min.   : 1.000   Catholic             : 151
##  Houston            : 137   1st Qu.: 5.000   CHD                  : 238
##  Northern California: 259   Median : 6.000   Private non-Christian: 145
##  Other              :1076   Mean   : 6.925   PUBLIC               :1705
##  Pacific Northwest  : 189   3rd Qu.:10.000
##  Southern California: 420   Max.   :11.000
##                             NA's   :2
##  Parent.Meeting.Flag Total.School.Enrollment  Income.Level EZ.Pay.Take.Up.Rate
##  0: 306              Min.   :  19.0          Q      :263   Min.   :0.0000
##  1:1933              1st Qu.: 353.2          O      :250   1st Qu.:0.1050
##                      Median : 594.0          L      :204   Median :0.2000
##                      Mean   : 641.4          P      :194   Mean   :0.2107
##                      3rd Qu.: 819.8          M      :182   3rd Qu.:0.2940
##                      Max.   :3990.0          J      :153   Max.   :1.7500
##                      NA's   :77              (Other):993
##  School.Sponsor      SPR.Product.Type SPR.New.Existing      FPP
##  0:1992         CA History   : 171    EXISTING:1543    Min.   :  2.00
##  1: 247         Costa Rica   :  44    NEW     : 696    1st Qu.: 12.00
##                 East Coast   :1884                     Median : 23.00
##                 IL History   :   1                     Mean   : 31.13
##                 International:   3                     3rd Qu.: 40.00
##                 Science      : 136                     Max.   :286.00
##
##  SPR.Group.Revenue NumberOfMeetingswithParents DifferenceTraveltoFirstMeeting
##  Min.   :  79      0: 306                      Min.   :-204
##  1st Qu.:1203      1:1379                      1st Qu.: 209
##  Median :1705      2: 554                      Median : 251
##  Mean   :1618                                  Mean   : 263
##  3rd Qu.:2049                                  3rd Qu.: 290
##  Max.   :4200                                  Max.   : 749
##                                                NA's   :306
##  DifferenceTraveltoLastMeeting  DepartureMonth MajorProgramCode
##  Min.   :-204.0                April   :510    C: 129
##  1st Qu.: 197.0                February: 45    H:1926
##  Median : 233.0                January :  9    I:   4
##  Mean   : 229.1                June    :838    S: 180
##  3rd Qu.: 261.0                March   :369
##  Max.   : 749.0                May     :468
##  NA's   :306
##  SingleGradeTripFlag FPP.to.School.enrollment   FPP.to.PAX
##  0: 909              Min.   :0.00092          Min.   :0.6000
##  1:1330              1st Qu.:0.02112          1st Qu.:0.8827
##                      Median :0.04542          Median :0.9091
##                      Mean   :0.06632          Mean   :0.9008
##                      3rd Qu.:0.08740          3rd Qu.:0.9333
##                      Max.   :2.05263          Max.   :1.0000
##                      NA's   :77
##  SchoolSizeIndicator Retained.in.2012. Deposit_month  grade_range
##     : 77             0: 874            10     :1310   E - E: 196
##  L  :553             1:1365            11     : 646   E - H:   2
##  M-L:564                               12     : 103   E - M:  59
##  S  :490                               9      :  76   H - H:  66
```

```
##  S-M:555                                6      :  51   M - H: 100
##                                         1      :  30   M - M:1816
##                                         (Other):  23
##    FRP_Actual       RPL_diff          MDR_range
##  Min.   :-22.00   Length:2239       M - M  :1088
##  1st Qu.:  4.00   Class :difftime   K - M  : 296
##  Median : 10.00   Mode  :numeric    PK - M : 255
##  Mean   : 13.53                     M - H  : 178
##  3rd Qu.: 19.00                     E - M  : 109
##  Max.   :147.00                     PK - H :  94
##                                     (Other): 219
```

```r
# Checking the category proportion of Target variable
df1 <- filter(dfn,dfn["Retained.in.2012."]==1)
df0 <- filter(dfn,dfn["Retained.in.2012."]==0)
print(paste("Count of Successful returns :",count(df0)))
```

```
## [1] "Count of Successful returns : 874"
```

```r
print(paste("Count of Failure to return :",count(df1)))
```

```
## [1] "Count of Failure to return : 1365"
```

```r
df2 <- dfn
df2$Retained.in.2012. <- as.factor(ifelse(df2$Retained.in.2012. == 0, "Fail", "Success"))

ggplot(df2, aes(x=factor(Retained.in.2012.)))+ geom_bar(stat="count", width=0.8,fill='springgreen3')+
  xlab('Did the group actually return') + ylab('Number of occurence')+
  ggtitle("Distribution of TARGET variable (SUCCESS & FAIL)")
```

EXPLORATION OF CATEGORICAL VARIABLES

```
print("Categorical variables in the dataset: ")
```

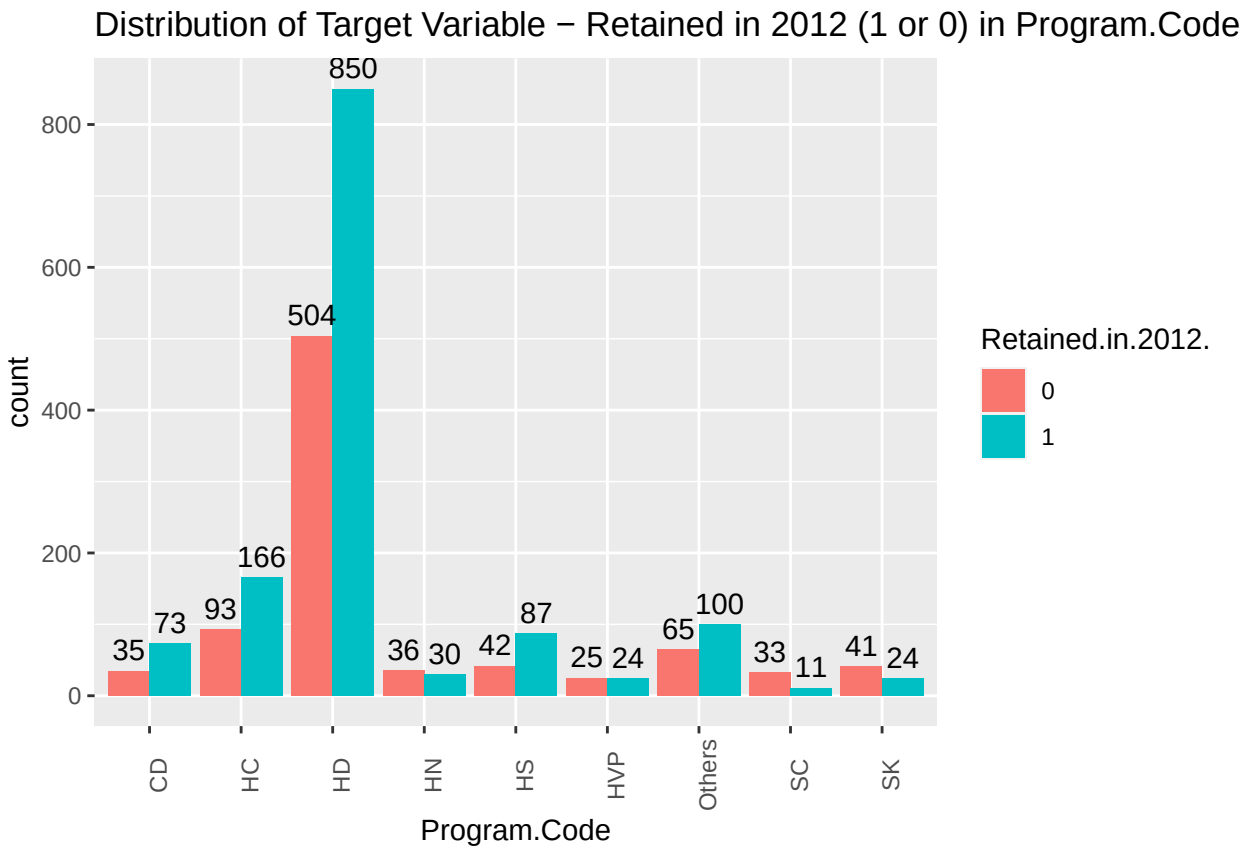
```
## [1] "Categorical variables in the dataset: "
```

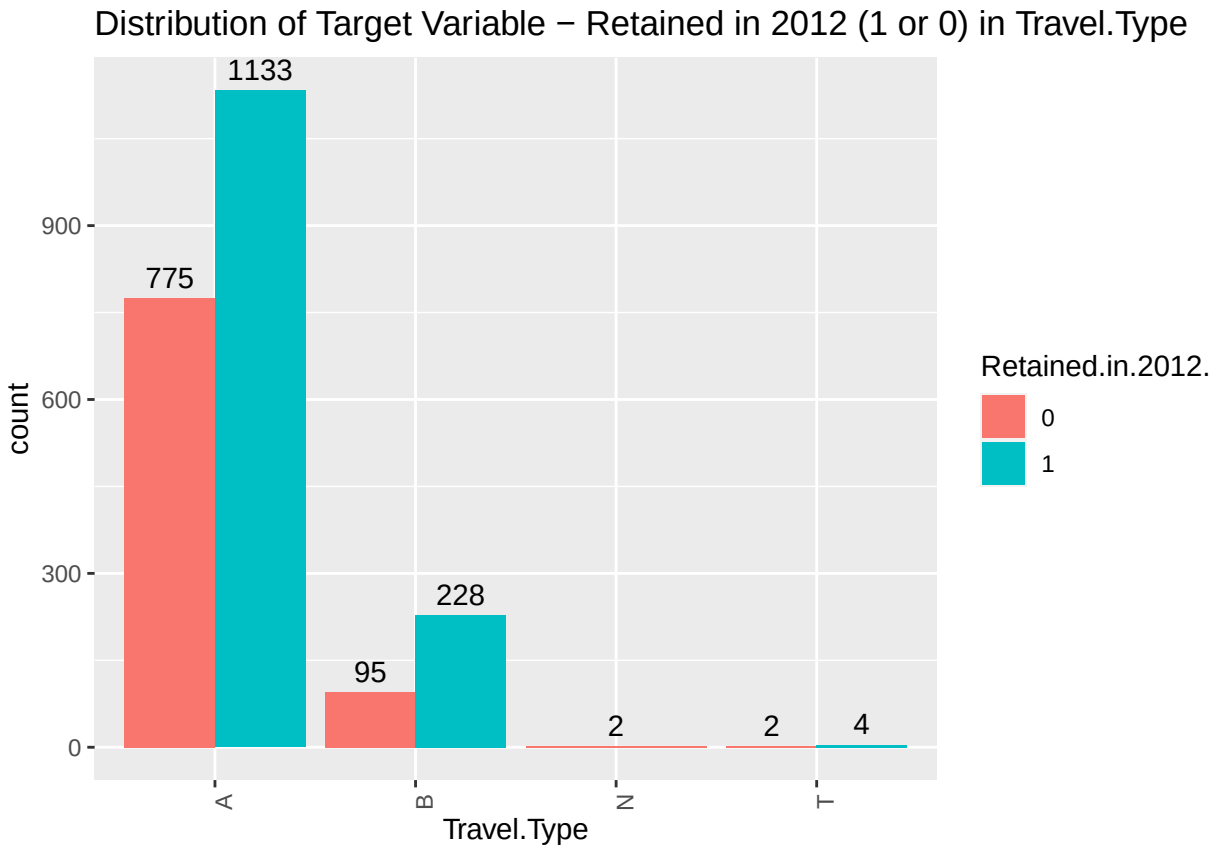
```
cat # displaying categorical columns
```

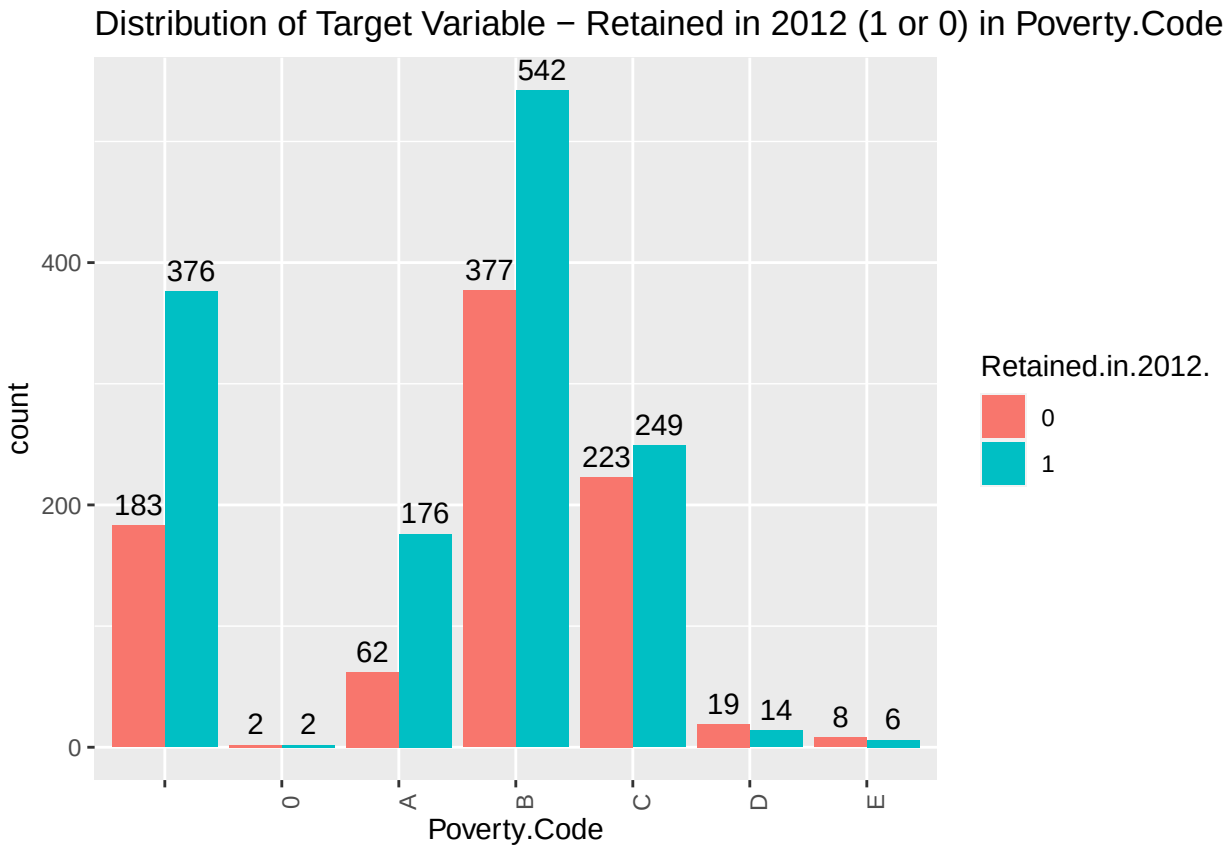
```
##  [1] "Program.Code"                "Travel.Type"
##  [3] "Poverty.Code"                "Region"
##  [5] "School.Type"                 "Income.Level"
##  [7] "SPR.Product.Type"            "SPR.New.Existing"
##  [9] "DepartureMonth"              "MajorProgramCode"
## [11] "SchoolSizeIndicator"         "grade_range"
## [13] "MDR_range"                   "Is.Non.Annual."
## [15] "Parent.Meeting.Flag"         "School.Sponsor"
## [17] "SingleGradeTripFlag"         "Retained.in.2012."
## [19] "Deposit_month"               "NumberOfMeetingswithParents"
```

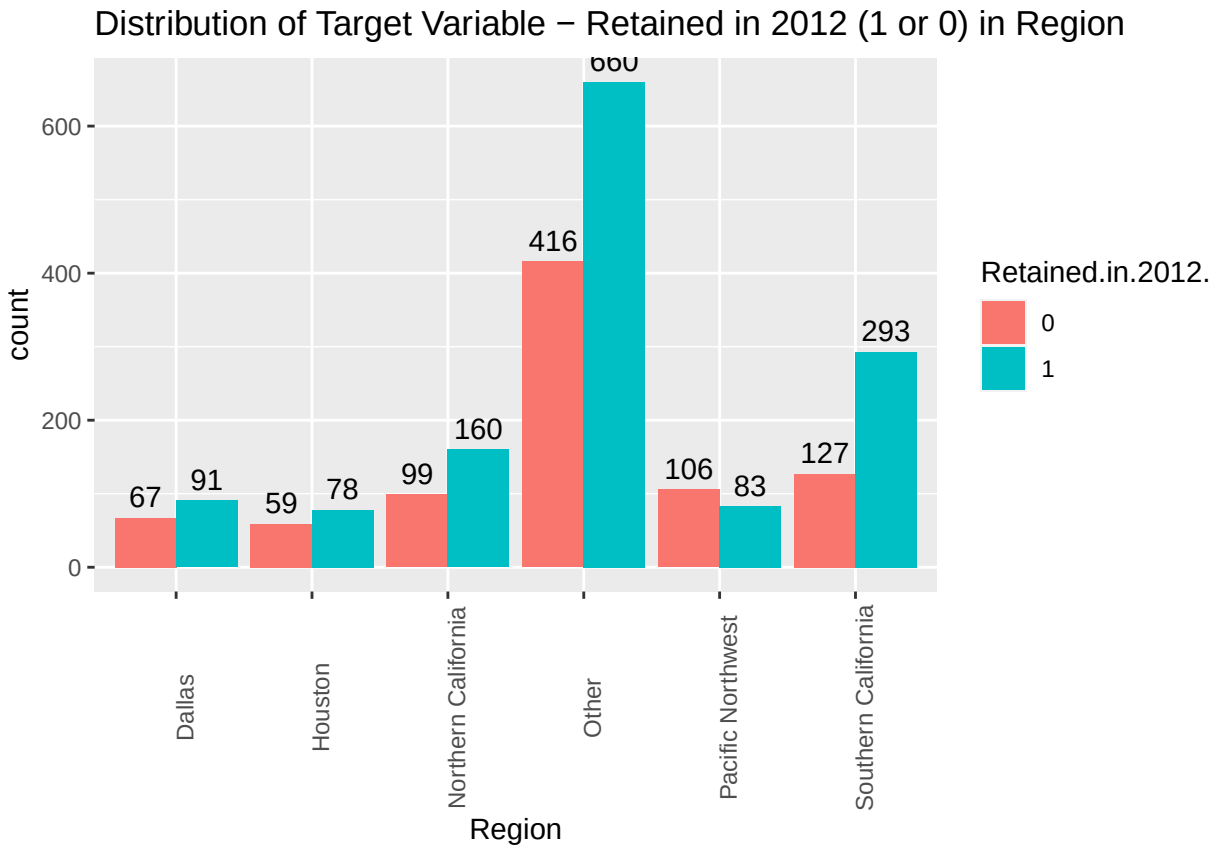
```
# Plotting all categorical variables with respect to Target variable
for (i in cat){
print(ggplot(dfn,aes_string(x=i, fill="Retained.in.2012."))+geom_bar(position="dodge")+
```

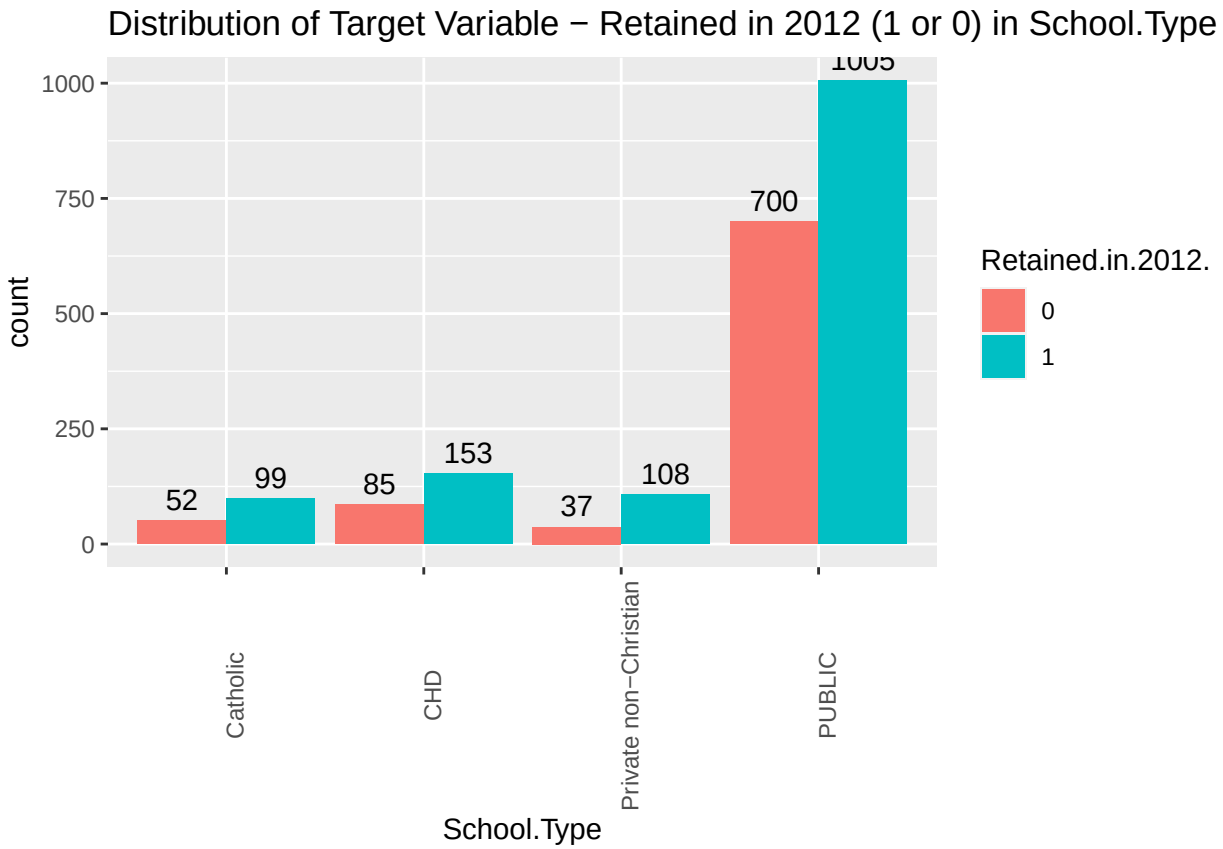
```
geom_text(stat='count', aes(label=..count..),position = position_dodge(0.9), vjust=-0.5)+
ggtitle(paste("Distribution of Target Variable - Retained in 2012 (1 or 0) in",i))+
theme(axis.text.x = element_text(angle = 90)))
}
```

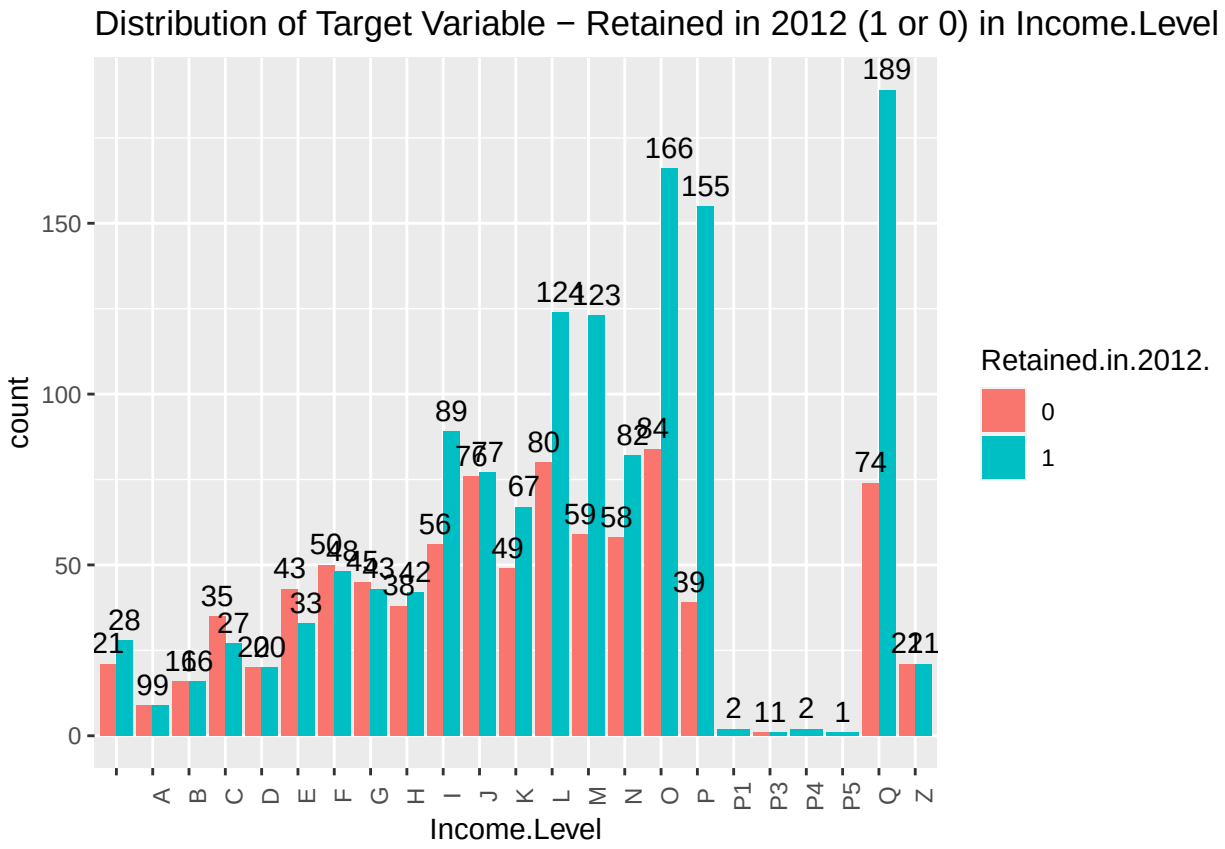


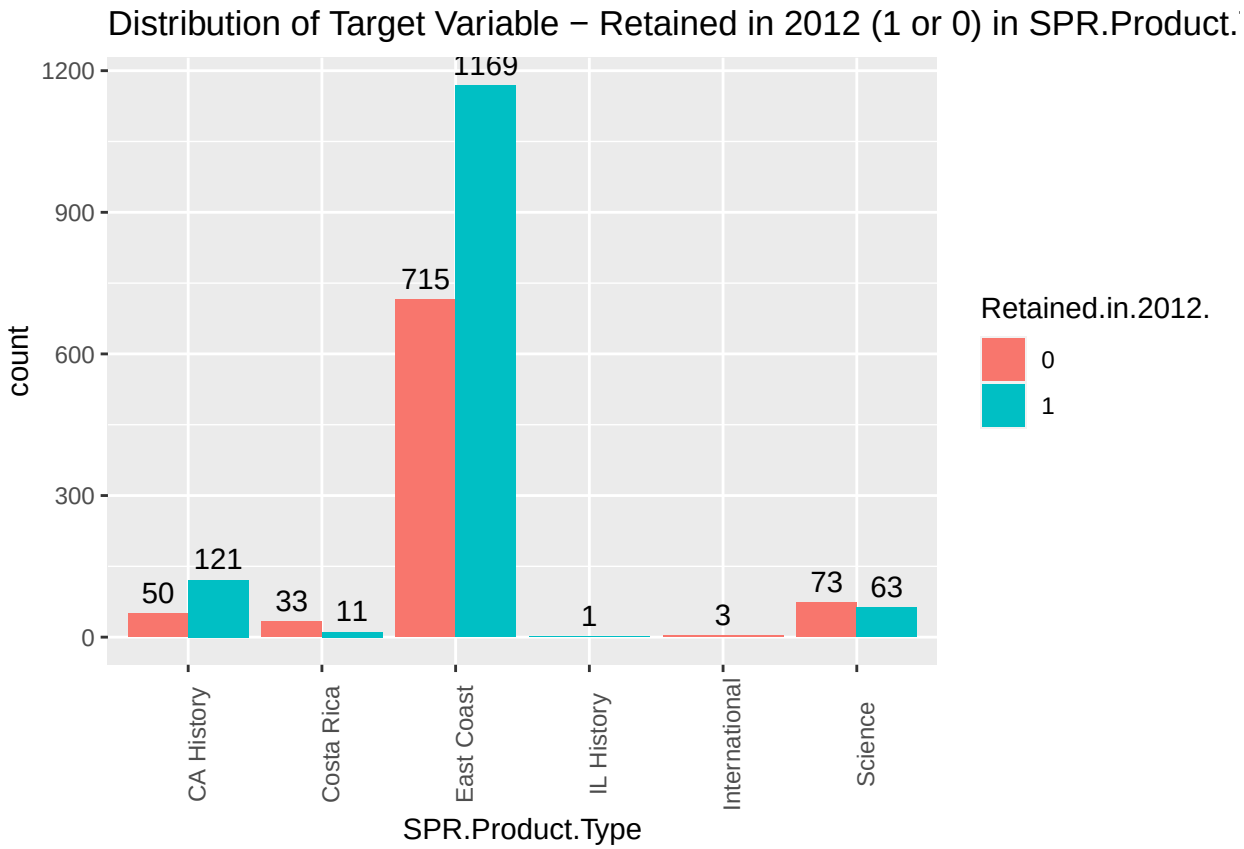

Distribution of Target Variable − Retained in 2012 (1 or 0) in Travel.Type
count
900
600
300
0
775
1133
95
228
2
2
4
A
B
N
T
Travel.Type
Retained.in.2012.
0
1

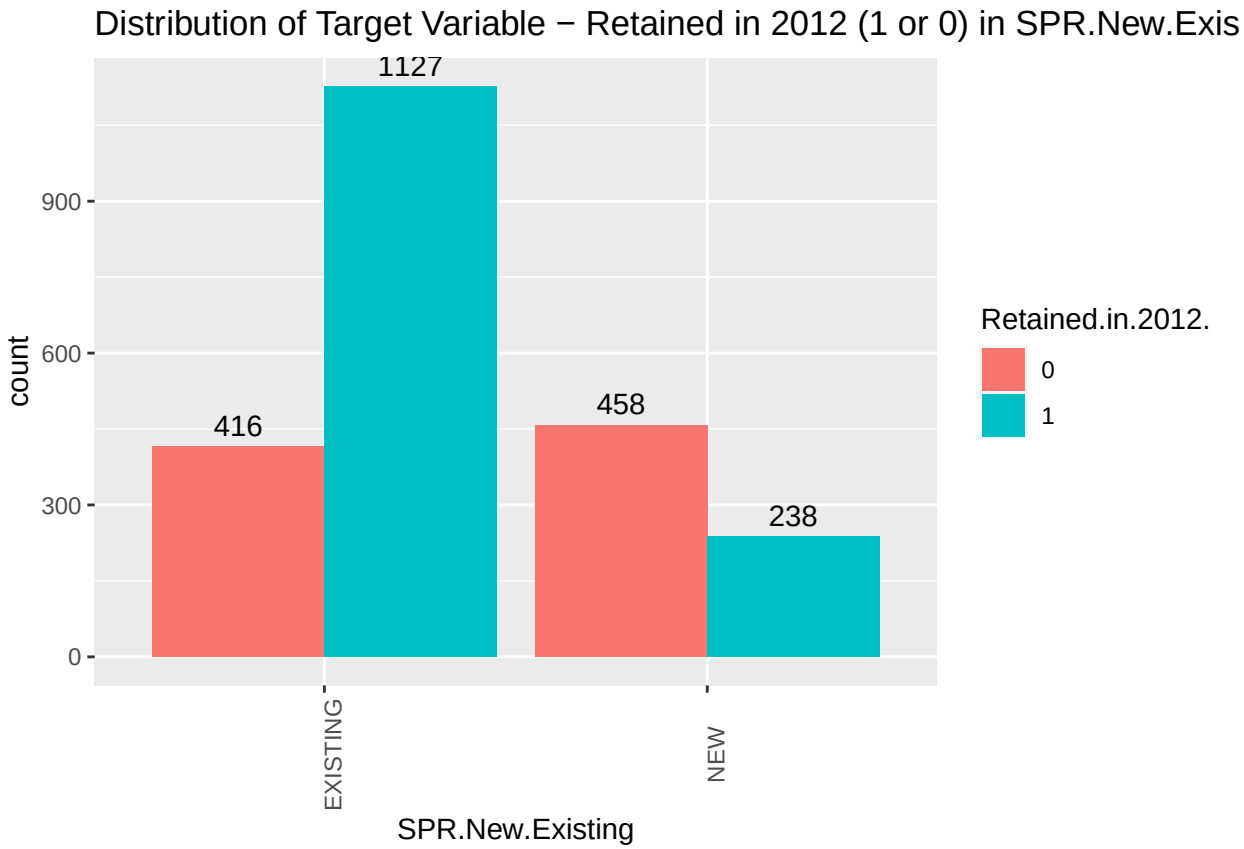

Distribution of Target Variable − Retained in 2012 (1 or 0) in Poverty.Code
count
400
200
0
183
376
2
2
62
176
377
542
223
249
19
14
8
6
0
A
B
C
D
E
Poverty.Code
Retained.in.2012.
0
1

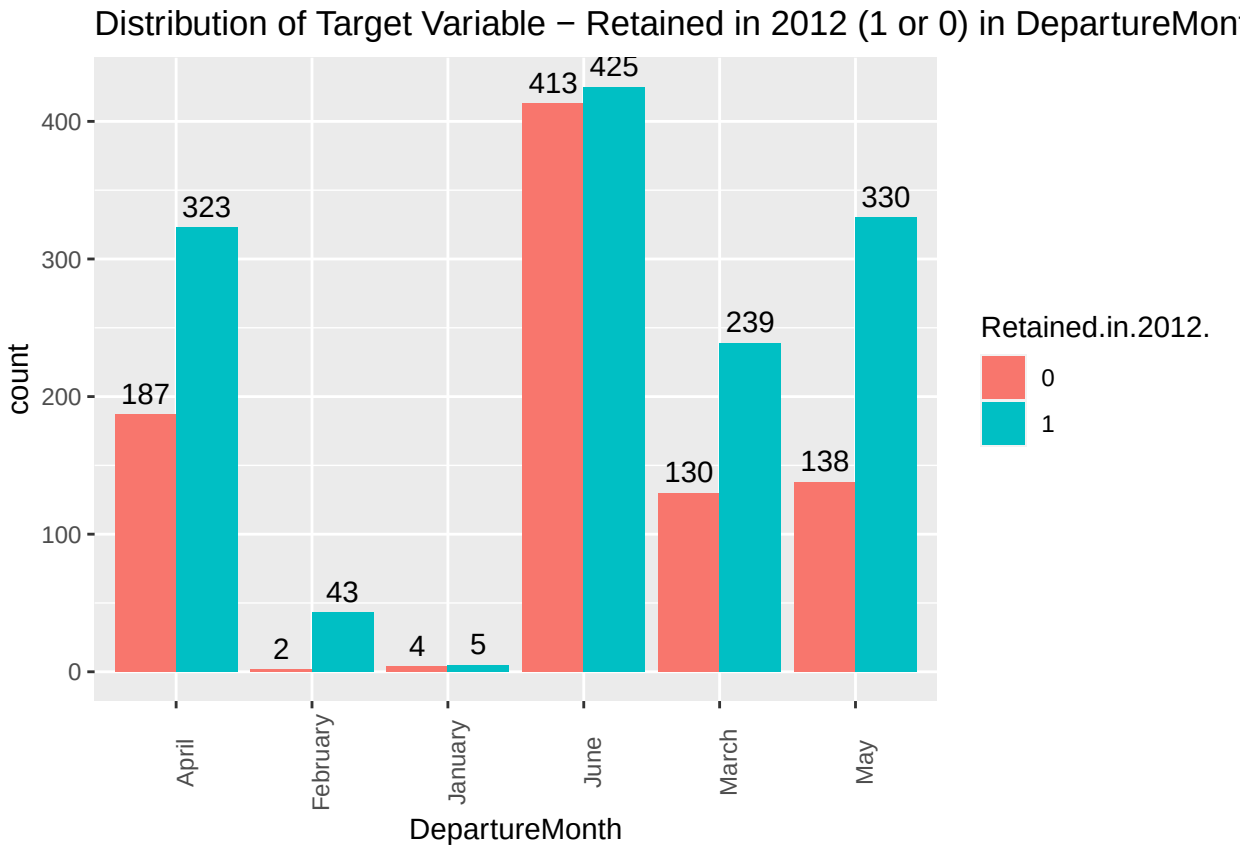

Distribution of Target Variable − Retained in 2012 (1 or 0) in Region
count
600
400
200
0
67
91
59
78
99
160
416
660
106
83
127
293
Dallas
Houston
Northern California
Other
Pacific Northwest
Southern California
Region
Retained.in.2012.
0
1

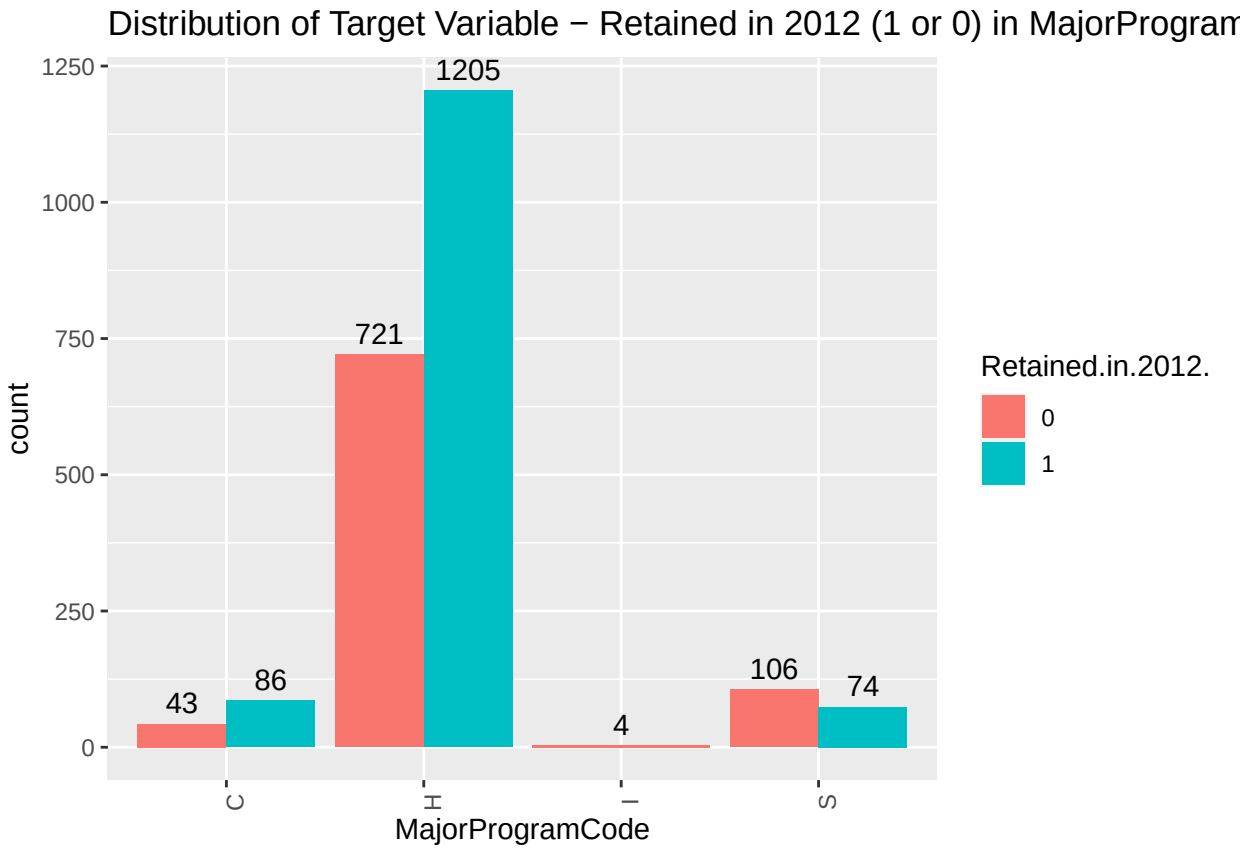

Distribution of Target Variable − Retained in 2012 (1 or 0) in School.Type
count
1000
750
500
250
0
52
99
85
153
37
108
700
1005
Catholic
CHD
Private non−Christian
PUBLIC
School.Type
Retained.in.2012.
0
1

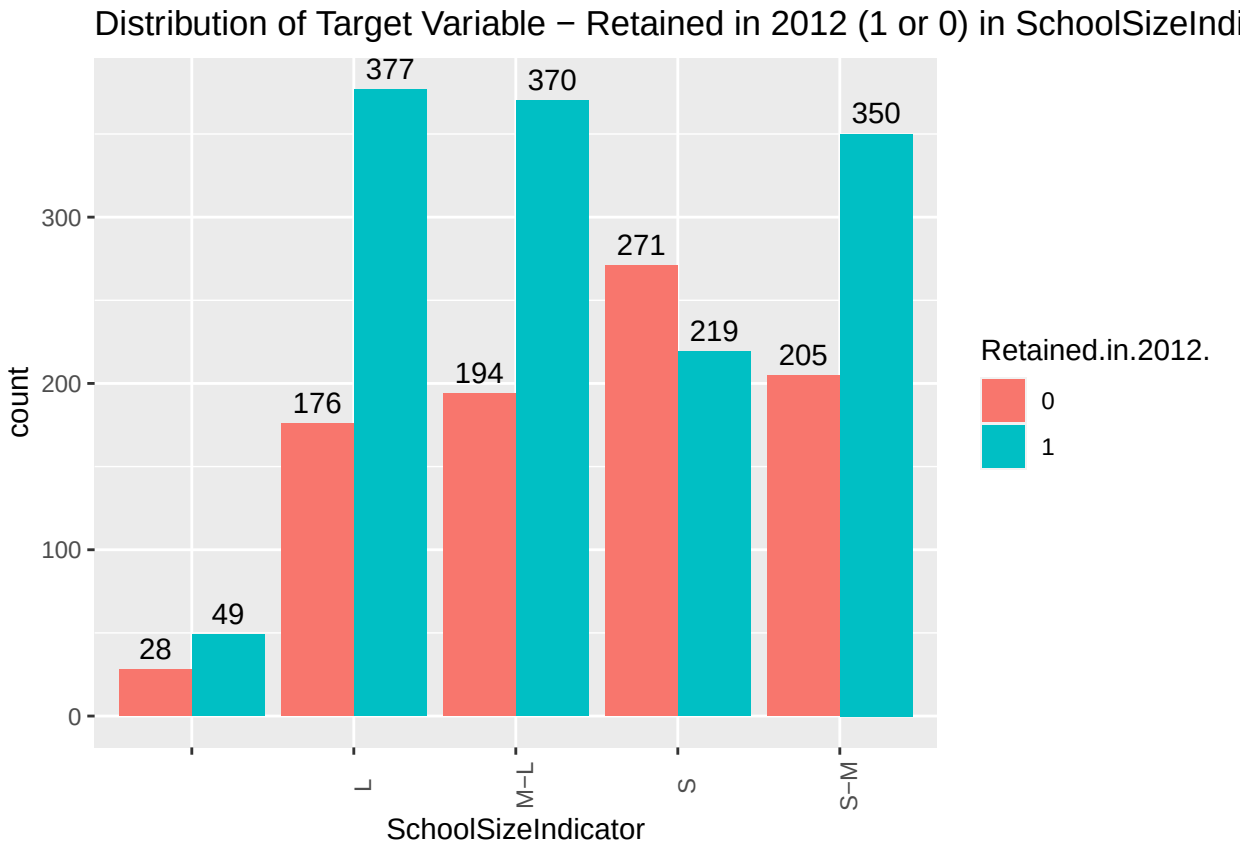

Distribution of Target Variable − Retained in 2012 (1 or 0) in Income.Level
count
Income.Level
Retained.in.2012.
0
1
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
P1
P3
P4
P5
Q
Z

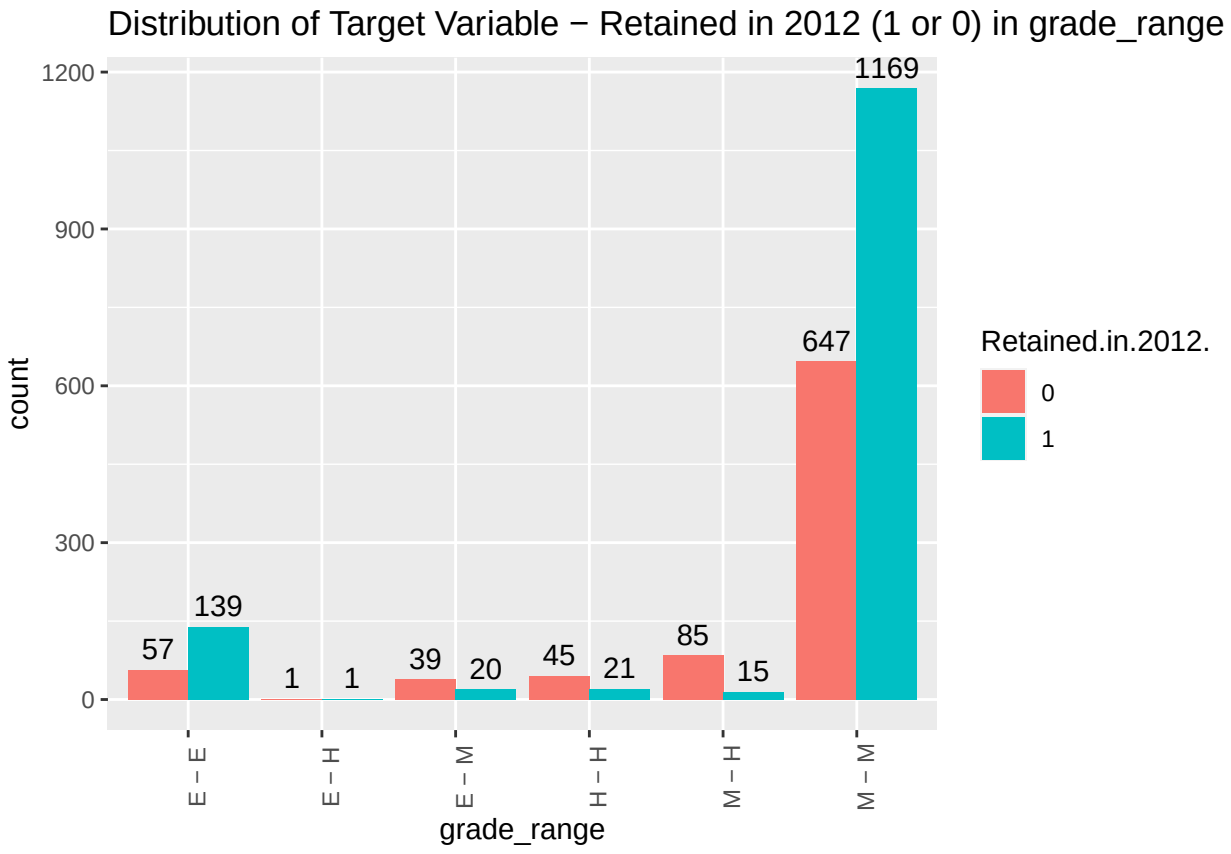

Distribution of Target Variable − Retained in 2012 (1 or 0) in SPR.Product.
count
1200
900
600
300
0
50
121
33
11
715
1169
1
3
73
63
CA History
Costa Rica
East Coast
IL History
International
Science
SPR.Product.Type
Retained.in.2012.
0
1

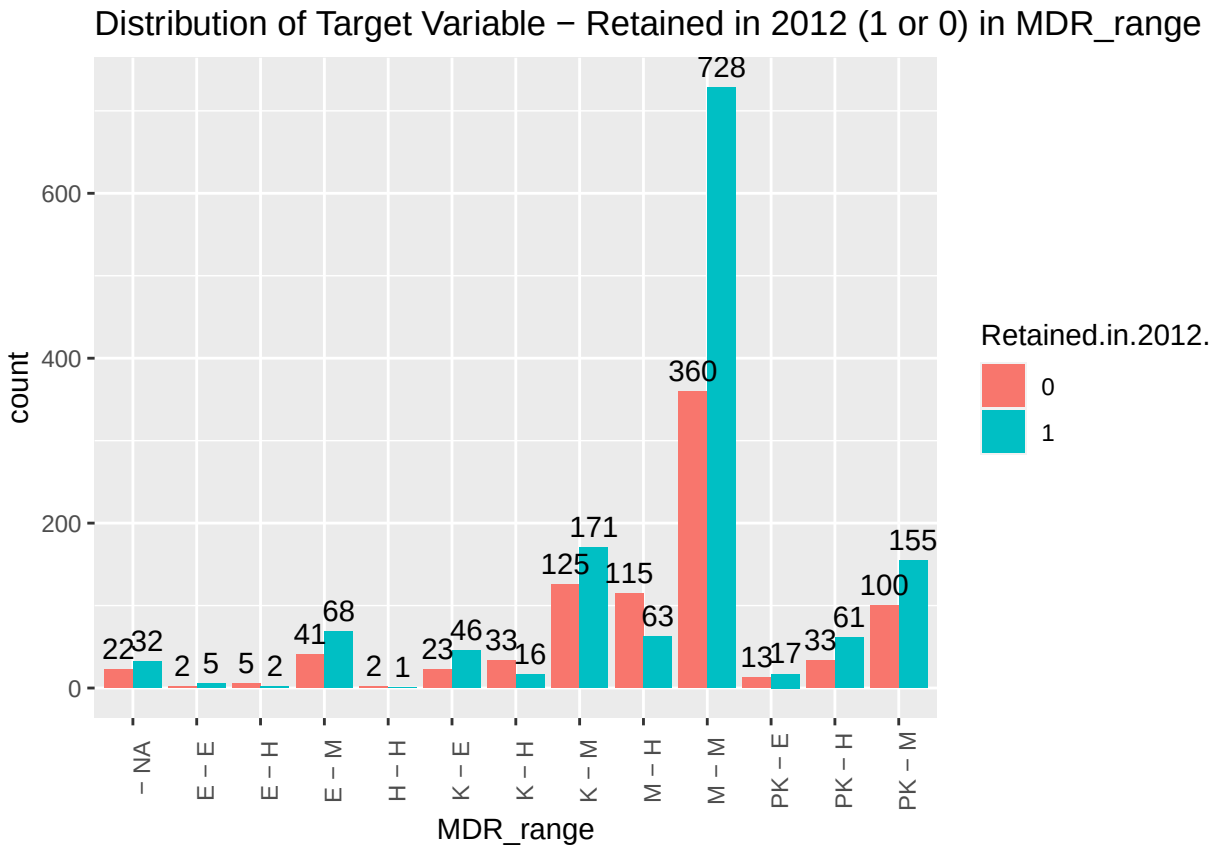

Distribution of Target Variable − Retained in 2012 (1 or 0) in SPR.New.Exis
1127
900
600
300
0
count
416
458
238
EXISTING
NEW
SPR.New.Existing
Retained.in.2012.
0
1

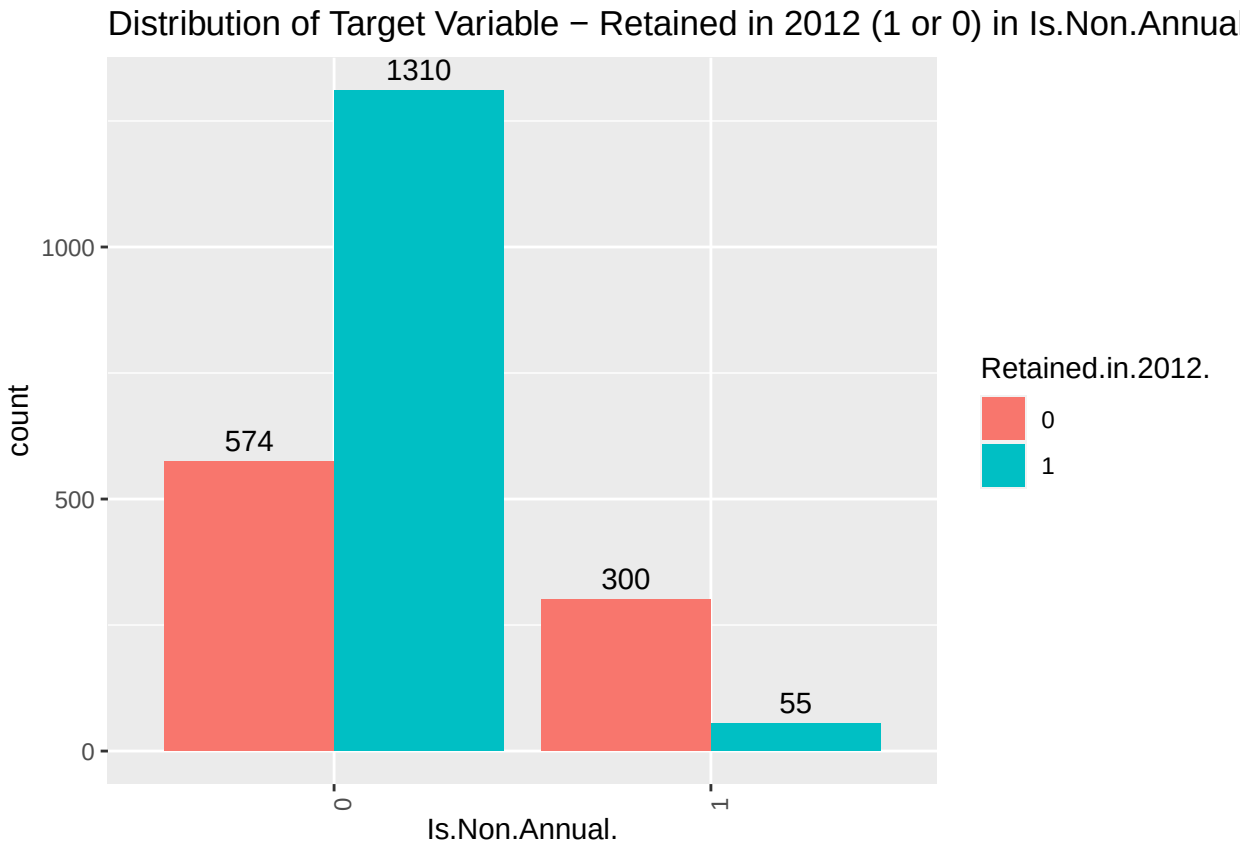

Distribution of Target Variable − Retained in 2012 (1 or 0) in DepartureMon
count
400
300
200
100
0
187
323
2
43
4
5
413
425
130
239
138
330
April
February
January
June
March
May
DepartureMonth
Retained.in.2012.
0
1

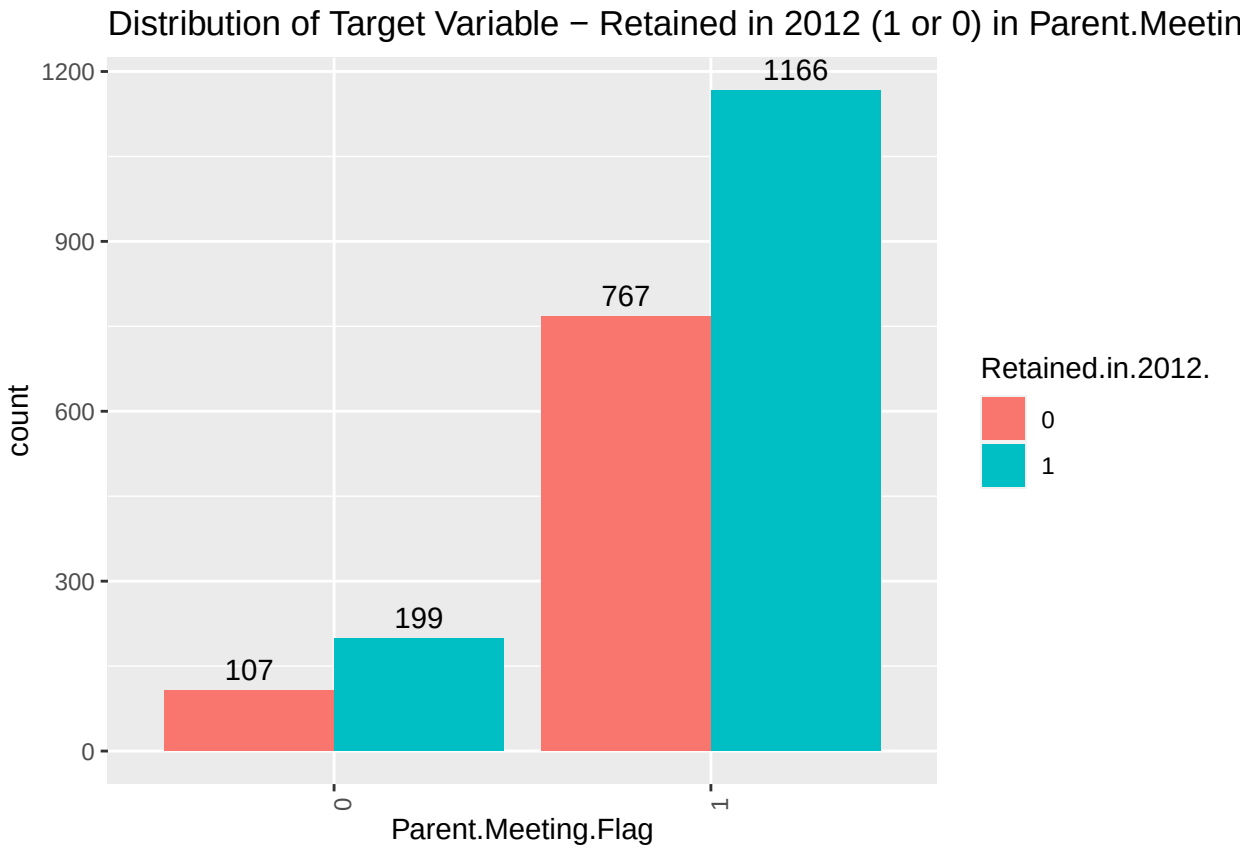

Distribution of Target Variable − Retained in 2012 (1 or 0) in MajorProgram
count
1250
1000
750
500
250
0
43
86
721
1205
4
106
74
C
H
I
S
MajorProgramCode
Retained.in.2012.
0
1

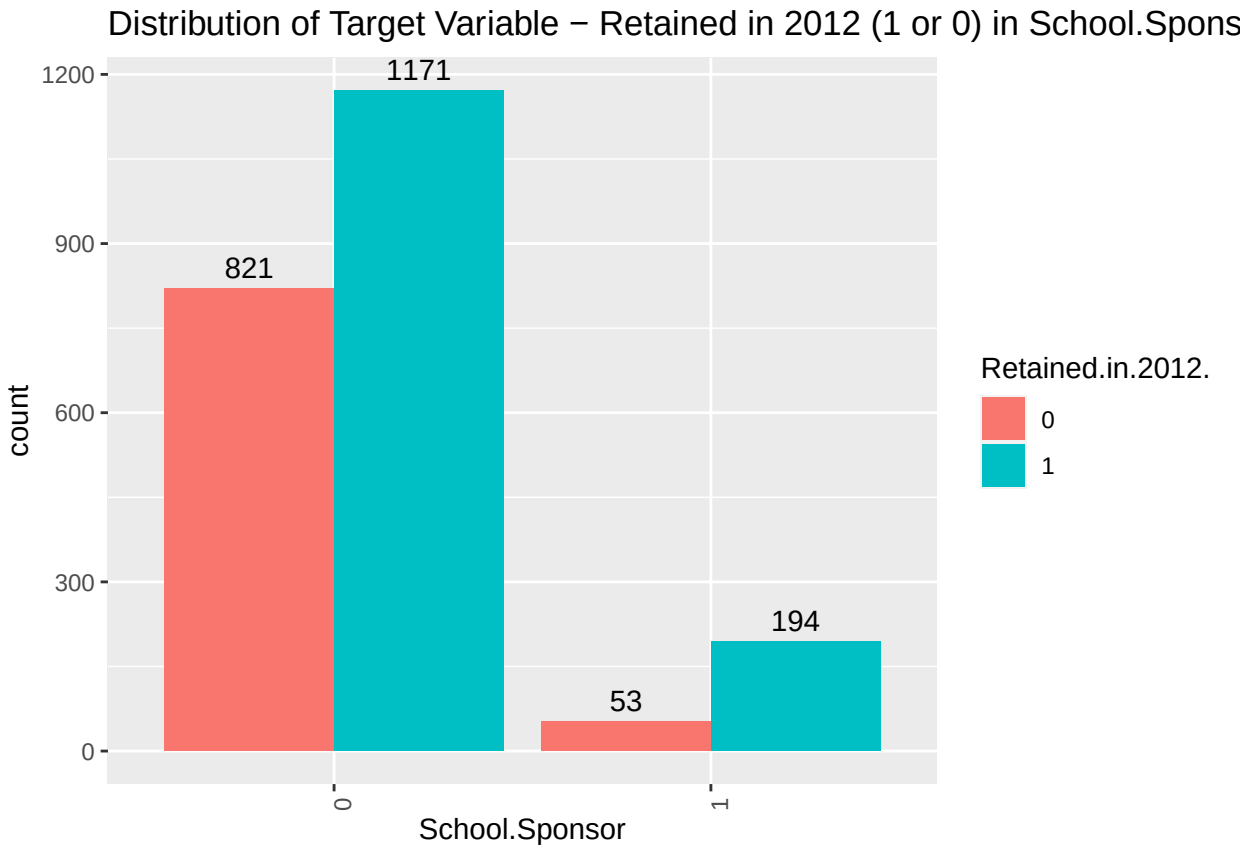

Distribution of Target Variable − Retained in 2012 (1 or 0) in SchoolSizeIndi
count
300
200
100
0
28
49
176
377
194
370
271
219
205
350
L
M−L
S
S−M
SchoolSizeIndicator
Retained.in.2012.
0
1

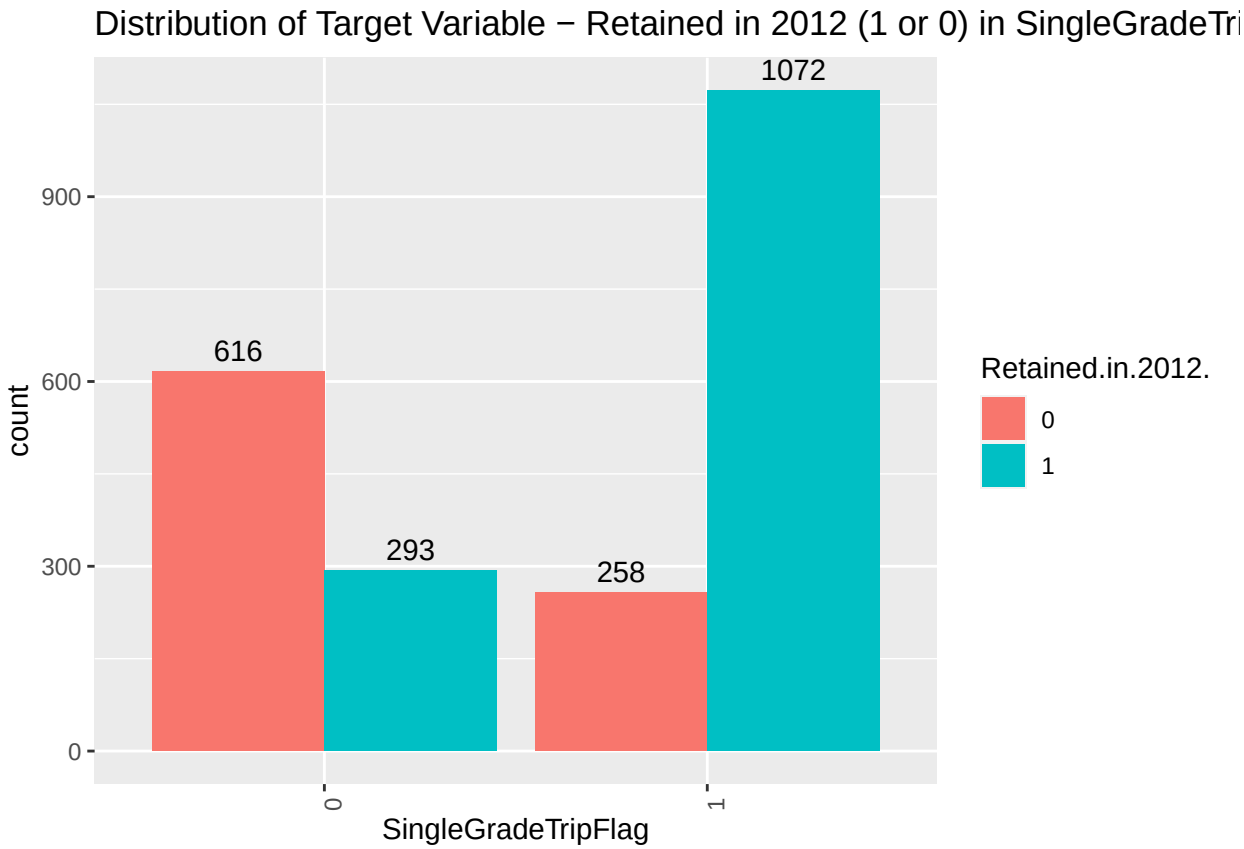

Distribution of Target Variable − Retained in 2012 (1 or 0) in grade_range
count
1200
900
600
300
0
57
139
1
1
39
20
45
21
85
15
647
1169
E − E
E − H
E − M
H − H
M − H
M − M
grade_range
Retained.in.2012.
0
1

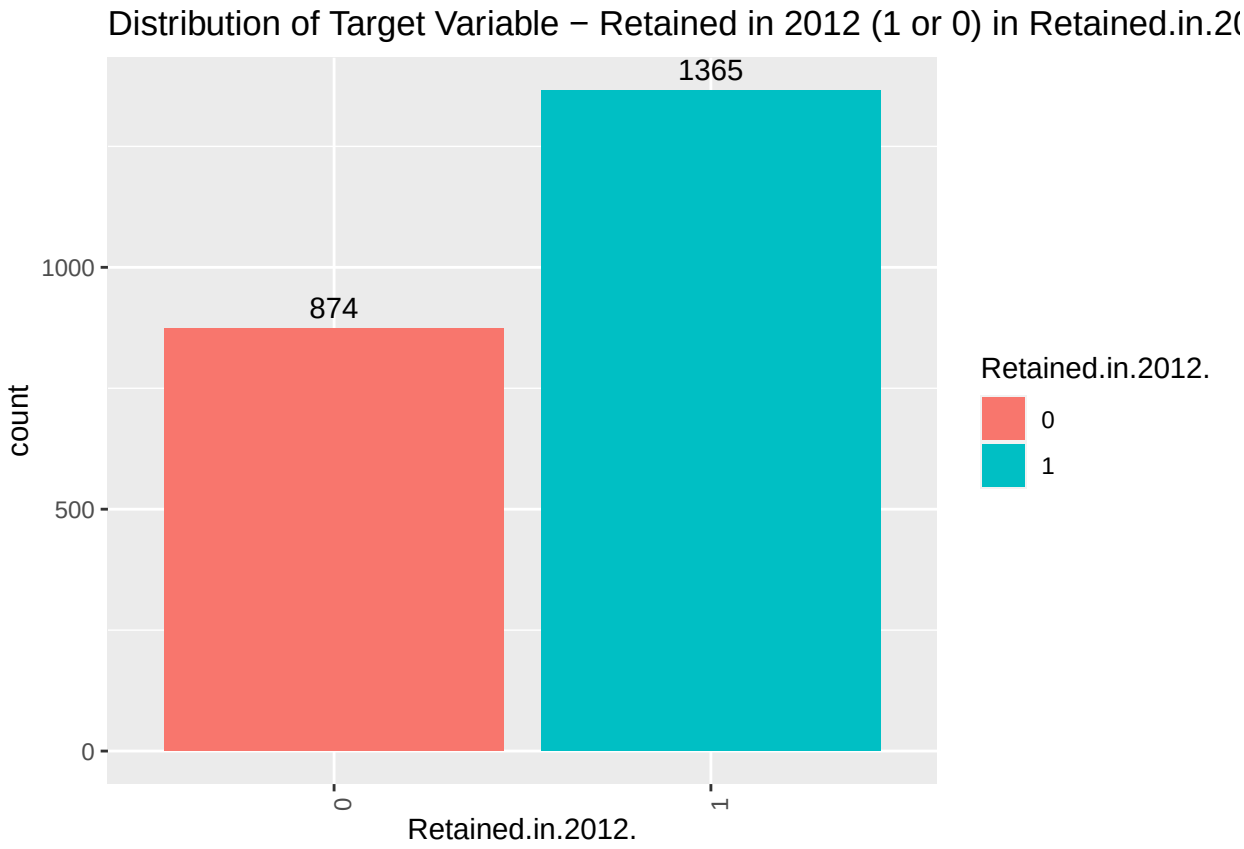

Distribution of Target Variable − Retained in 2012 (1 or 0) in MDR_range
count
0
200
400
600
22
32
2
5
5
2
41
68
2
1
23
46
33
16
125
171
115
63
360
728
13
17
33
61
100
155
− NA
E − E
E − H
E − M
H − H
K − E
K − H
K − M
M − H
M − M
PK − E
PK − H
PK − M
MDR_range
Retained.in.2012.
0
1

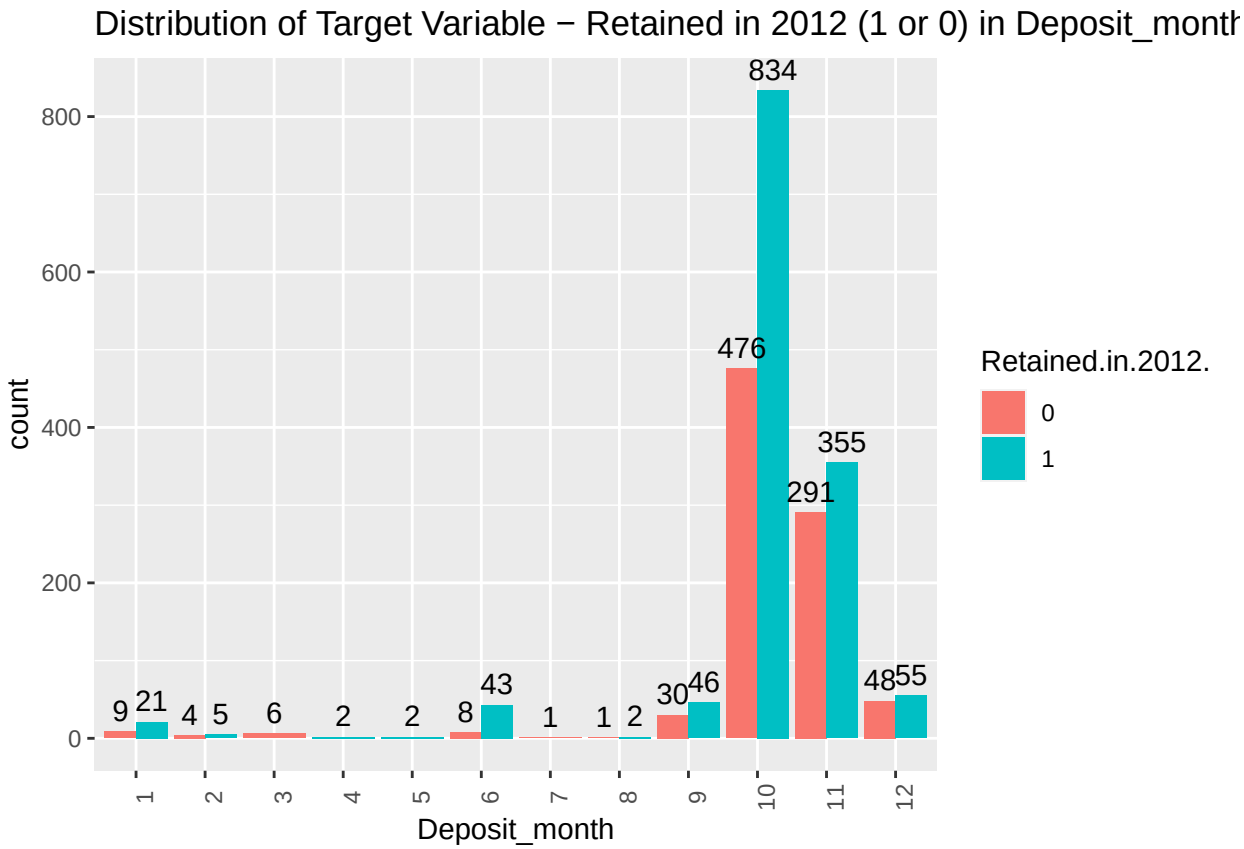

Distribution of Target Variable − Retained in 2012 (1 or 0) in Is.Non.Annua
1310
574
300
55
1000
500
0
count
0
1
Is.Non.Annual.
Retained.in.2012.
0
1

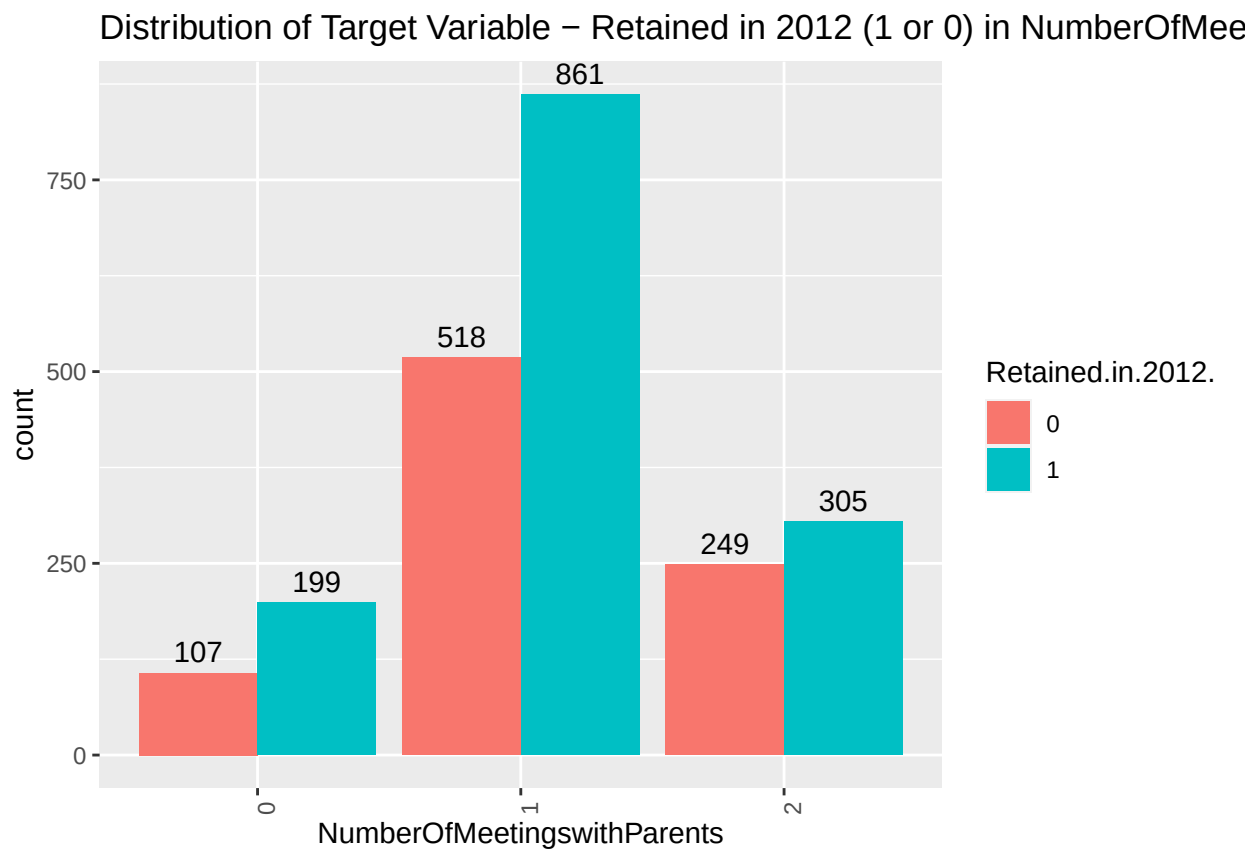

Distribution of Target Variable − Retained in 2012 (1 or 0) in Parent.Meetin
count
1200
900
600
300
0
107
199
767
1166
0
1
Parent.Meeting.Flag
Retained.in.2012.
0
1


Distribution of Target Variable − Retained in 2012 (1 or 0) in School.Spons
1200
900
600
300
0
count
1171
821
194
53
0
1
School.Sponsor
Retained.in.2012.
0
1


Distribution of Target Variable − Retained in 2012 (1 or 0) in SingleGradeTri
count
900
600
300
0
616
293
258
1072
0
1
SingleGradeTripFlag
Retained.in.2012.
0
1

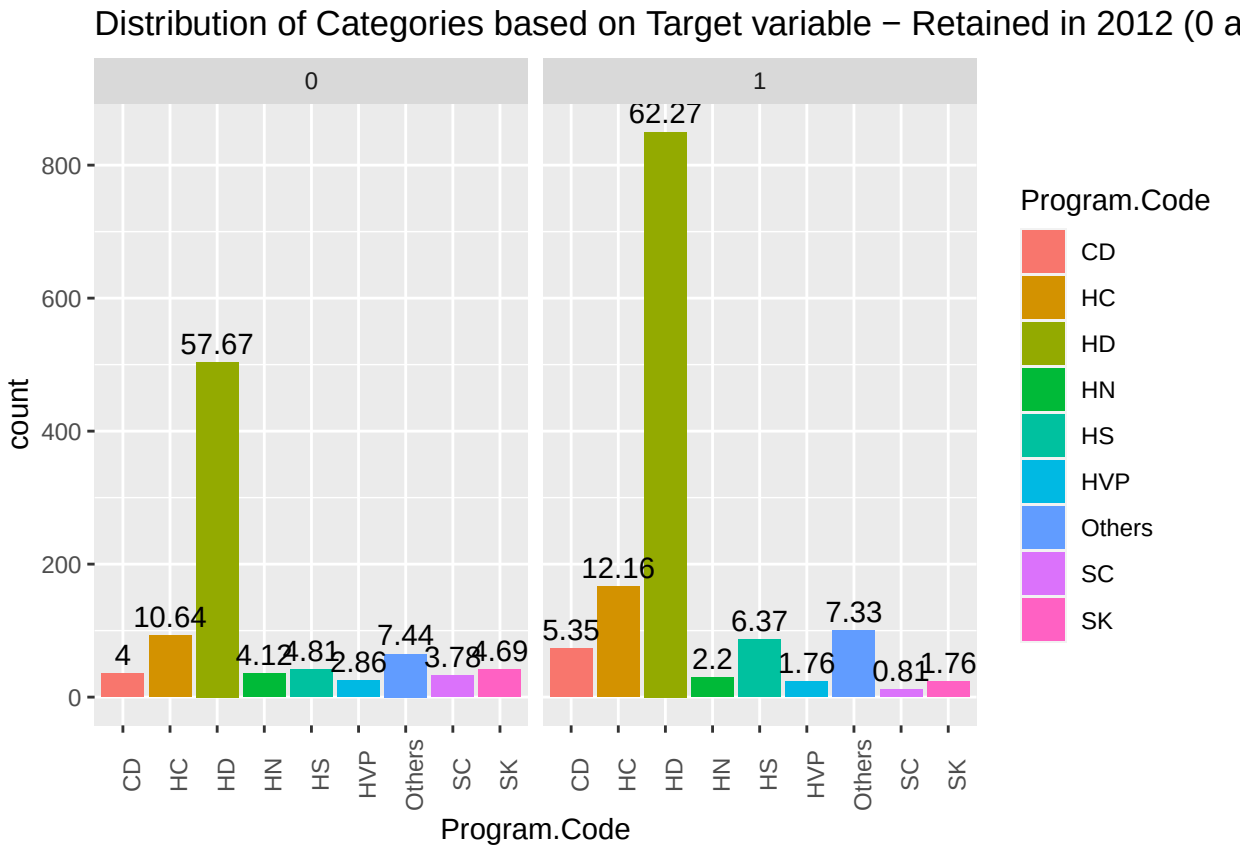

Distribution of Target Variable − Retained in 2012 (1 or 0) in Retained.in.2(
1365
874
1000
500
0
count
0
1
Retained.in.2012.
Retained.in.2012.
0
1

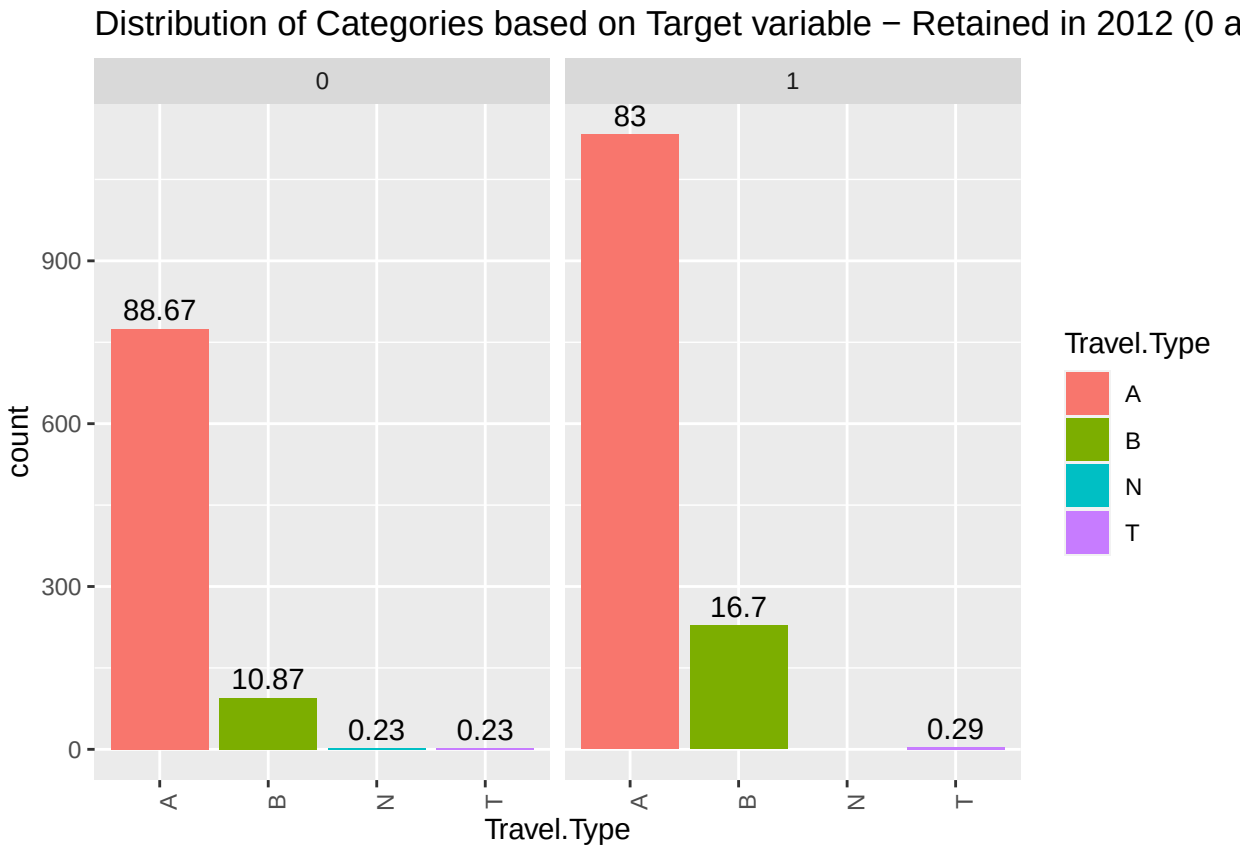

Distribution of Target Variable − Retained in 2012 (1 or 0) in Deposit_month
count
800
600
400
200
0
9
21
4
5
6
2
2
8
43
1
1
2
30
46
476
834
291
355
48
55
1
2
3
4
5
6
7
8
9
10
11
12
Deposit_month
Retained.in.2012.
0
1

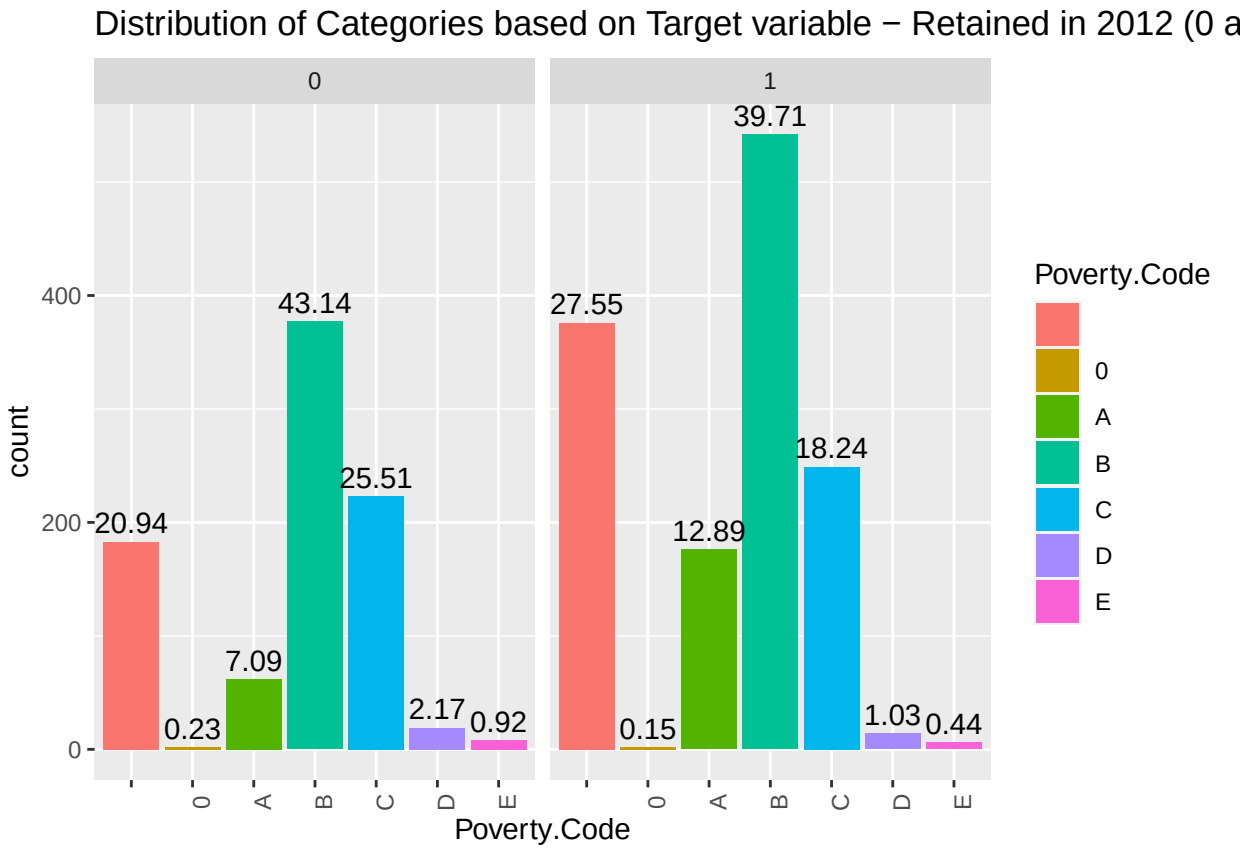


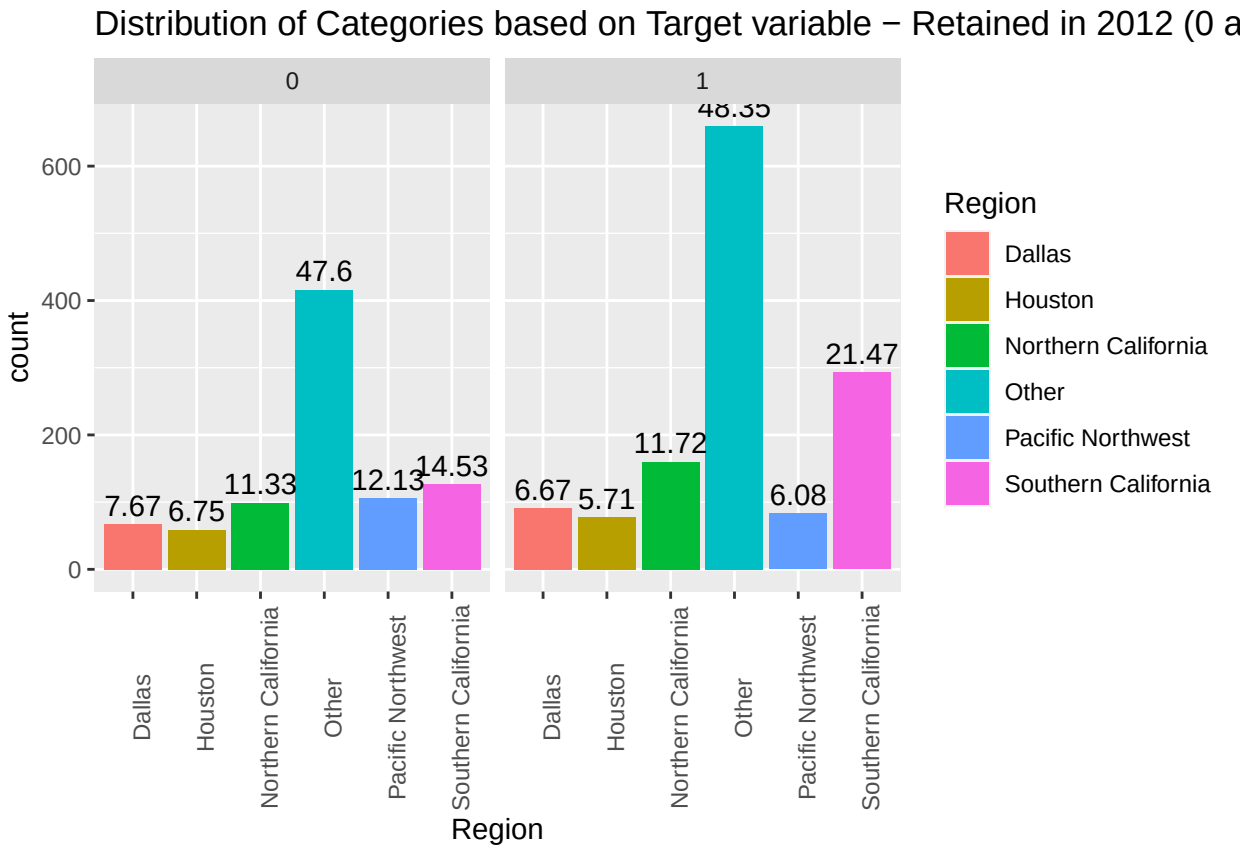
From the above plots we can check the proportion of 0 and 1 of Target in each category of the input variable. If the proportions are same across all categories then the input variable is not much significant in predicting the Target. In other words it won't affect our Target variable to a significant level.

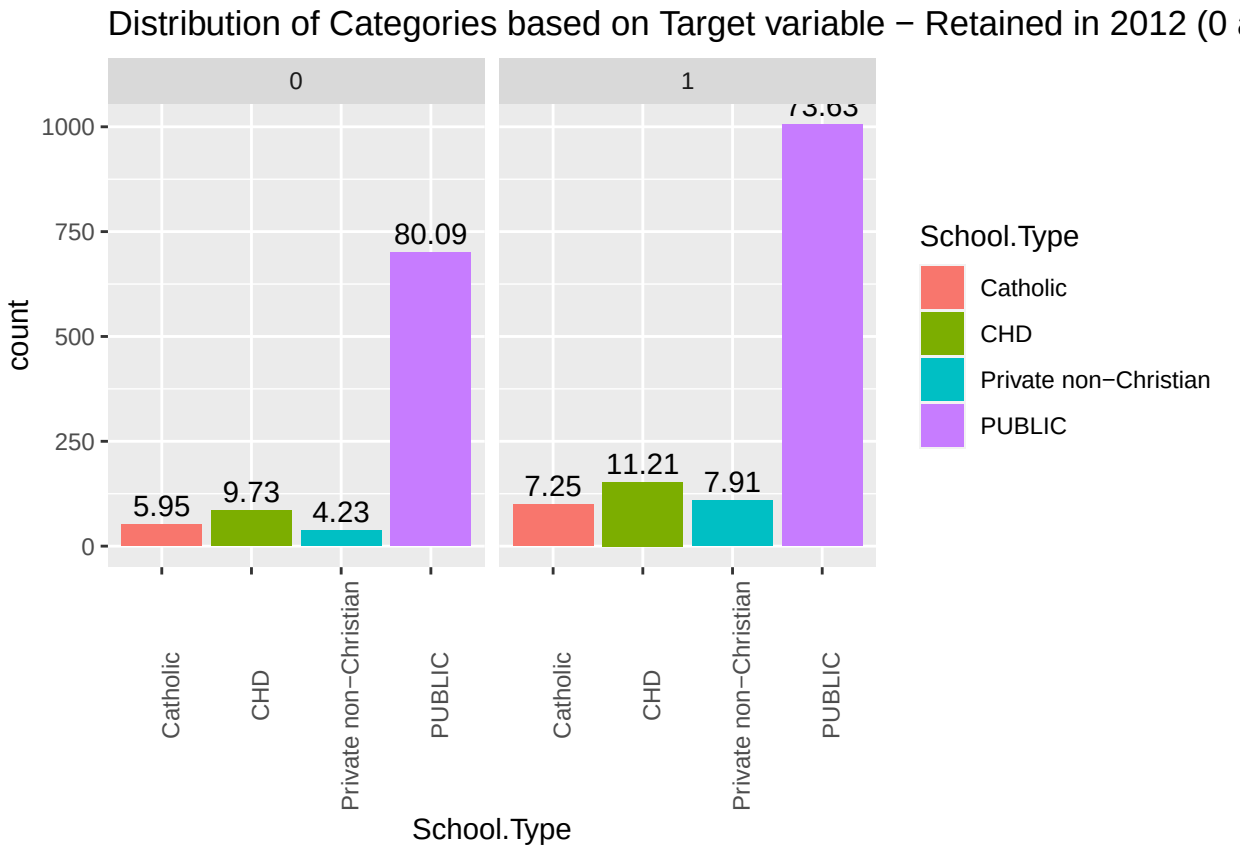
```
# Distribution of input variable categories in each Target category
for (i in cat){
print(ggplot(dfn,aes_string(x=i, fill=i))+geom_bar(position="dodge")+
geom_text(aes(label=round(after_stat(prop*100),2), group=1),stat='count',
size=4,position = position_dodge(0.9), vjust=-0.4) + facet_wrap('Retained.in.2012.')
+ ggtitle("Distribution of Categories based on Target variable - Retained in 2012 (0 and 1)")+
  theme(axis.text.x = element_text(angle = 90)))
}
```

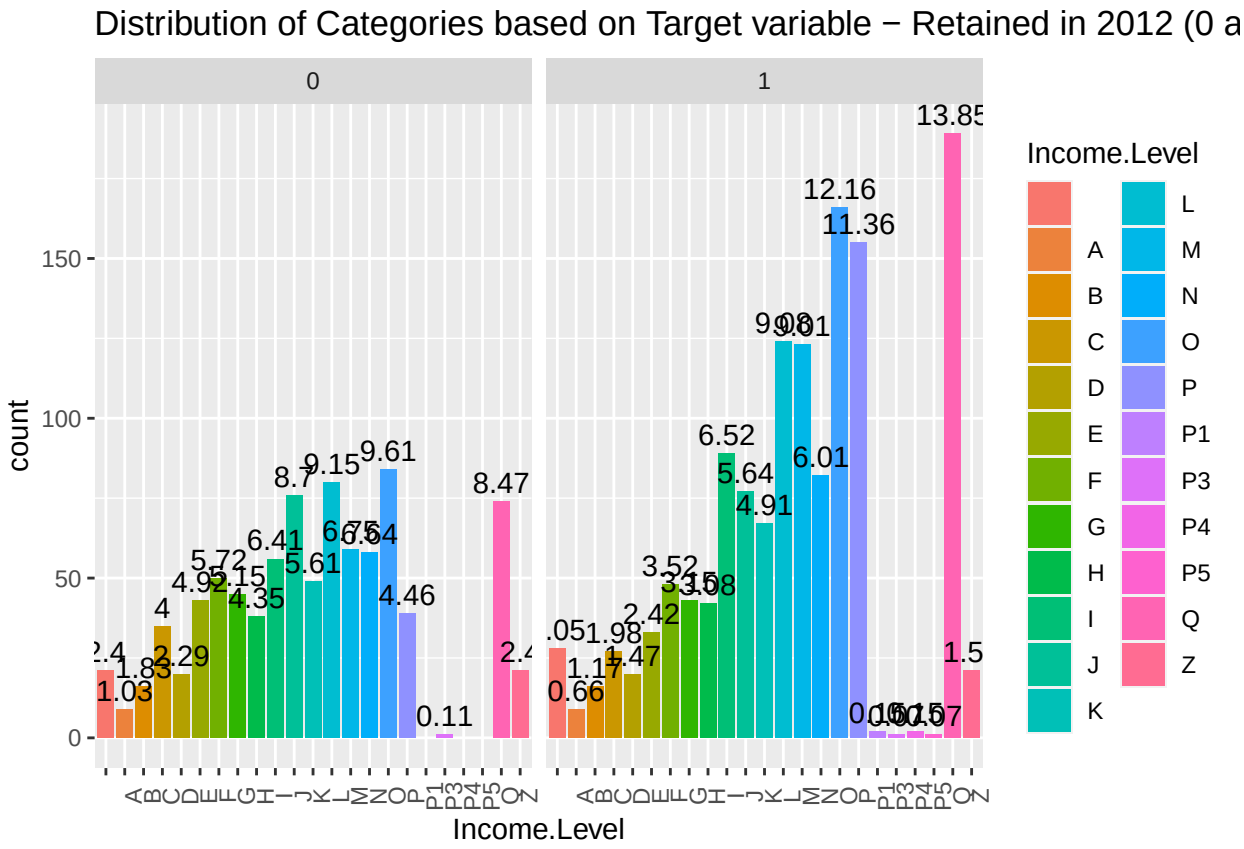

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
800
600
400
200
0
57.67
10.64
4
4.12
4.81
2.86
7.44
3.78
4.69
62.27
12.16
5.35
2.2
6.37
1.76
7.33
0.81
1.76
CD
HC
HD
HN
HS
HVP
Others
SC
SK
Program.Code
Program.Code
CD
HC
HD
HN
HS
HVP
Others
SC
SK

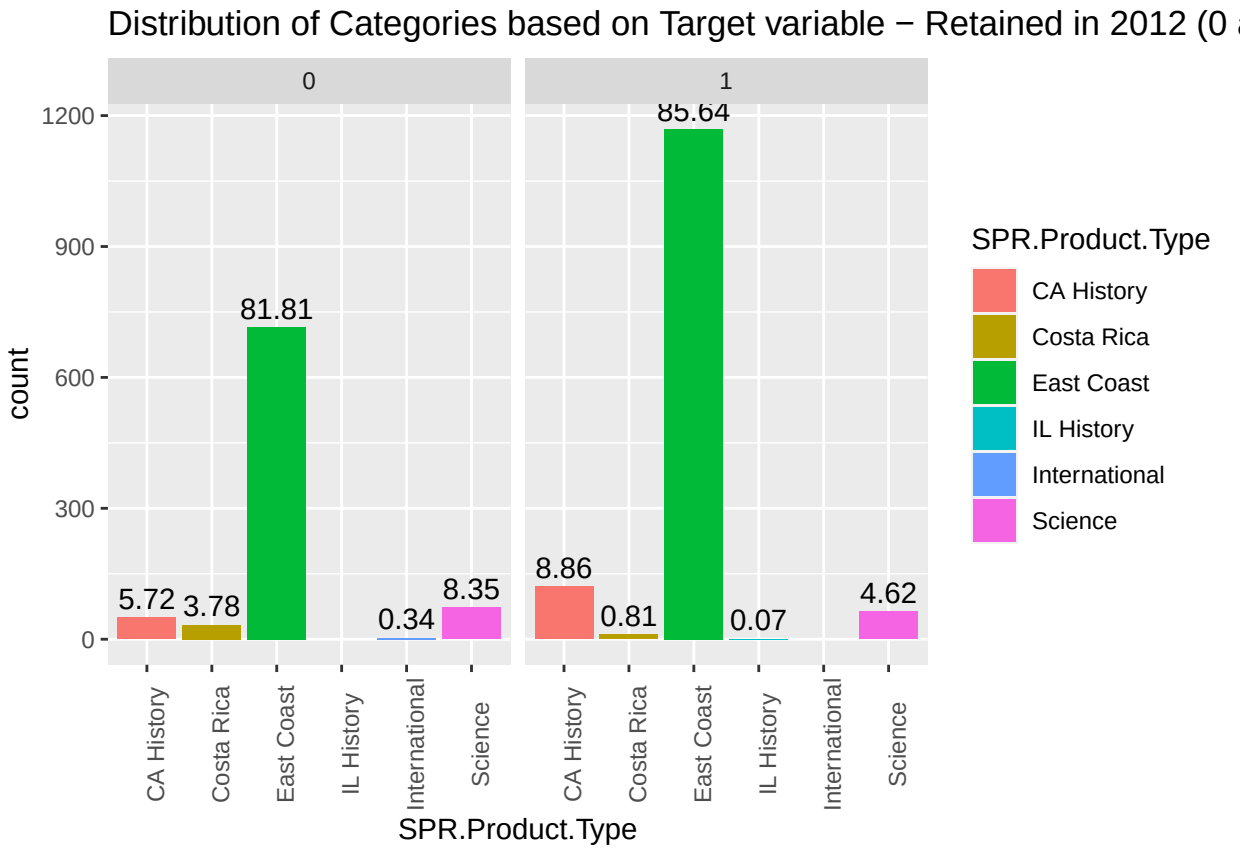

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
83
900
88.67
600
count
300
16.7
10.87
0.23
0.23
0.29
0
A
B
N
T
A
B
N
T
Travel.Type
Travel.Type
A
B
N
T

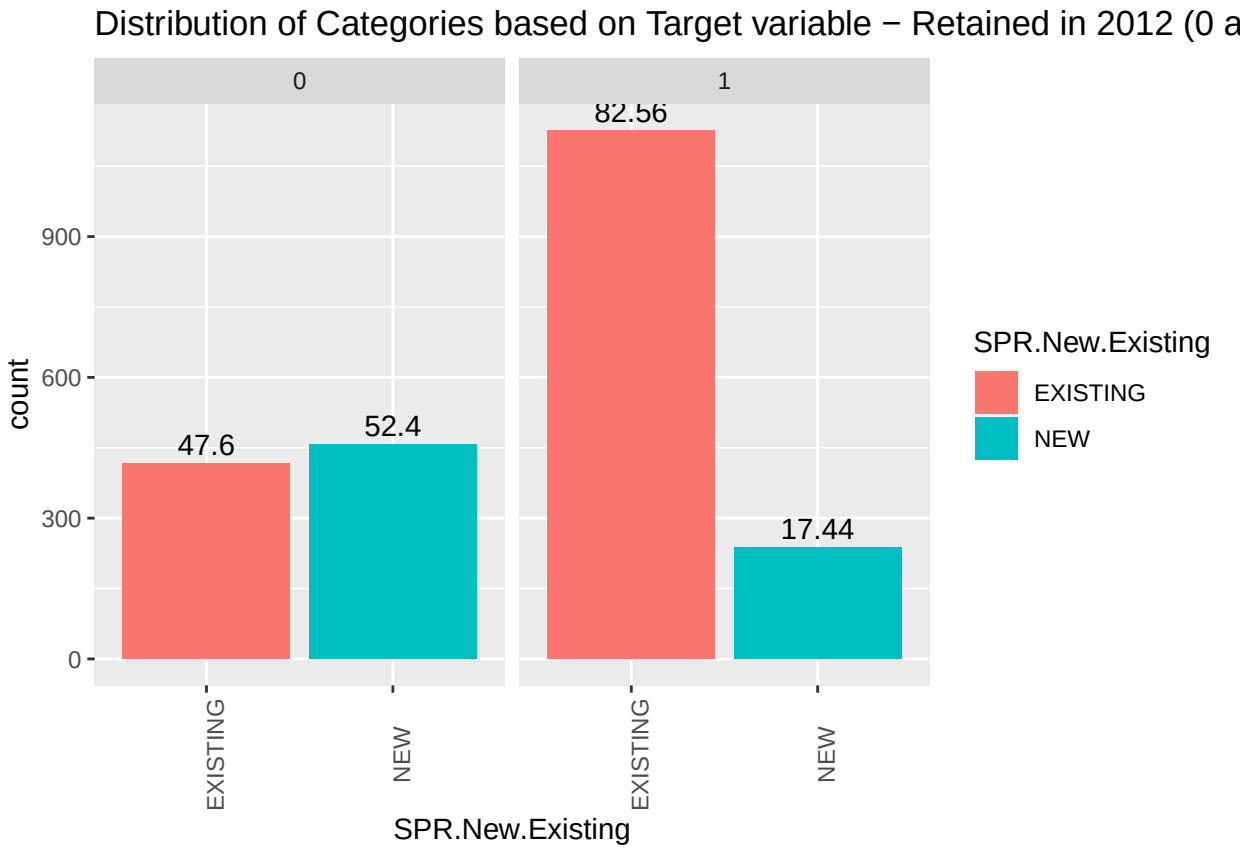

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
400
200
0
20.94
0.23
7.09
43.14
25.51
2.17
0.92
27.55
0.15
12.89
39.71
18.24
1.03
0.44
0
A
B
C
D
E
Poverty.Code
Poverty.Code
0
A
B
C
D
E

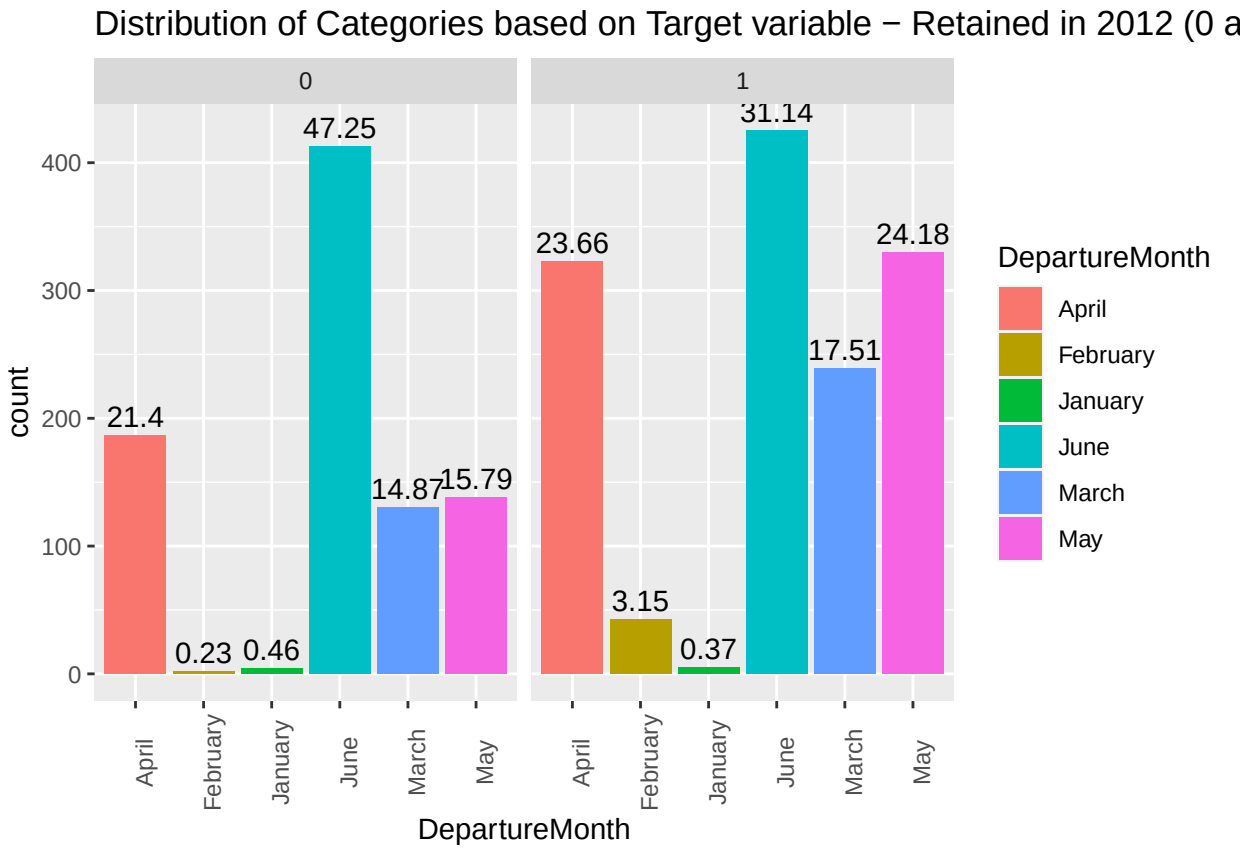

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
600
400
200
0
7.67
6.75
11.33
47.6
12.13
14.53
6.67
5.71
11.72
48.35
6.08
21.47
Dallas
Houston
Northern California
Other
Pacific Northwest
Southern California
Region
Region
Dallas
Houston
Northern California
Other
Pacific Northwest
Southern California

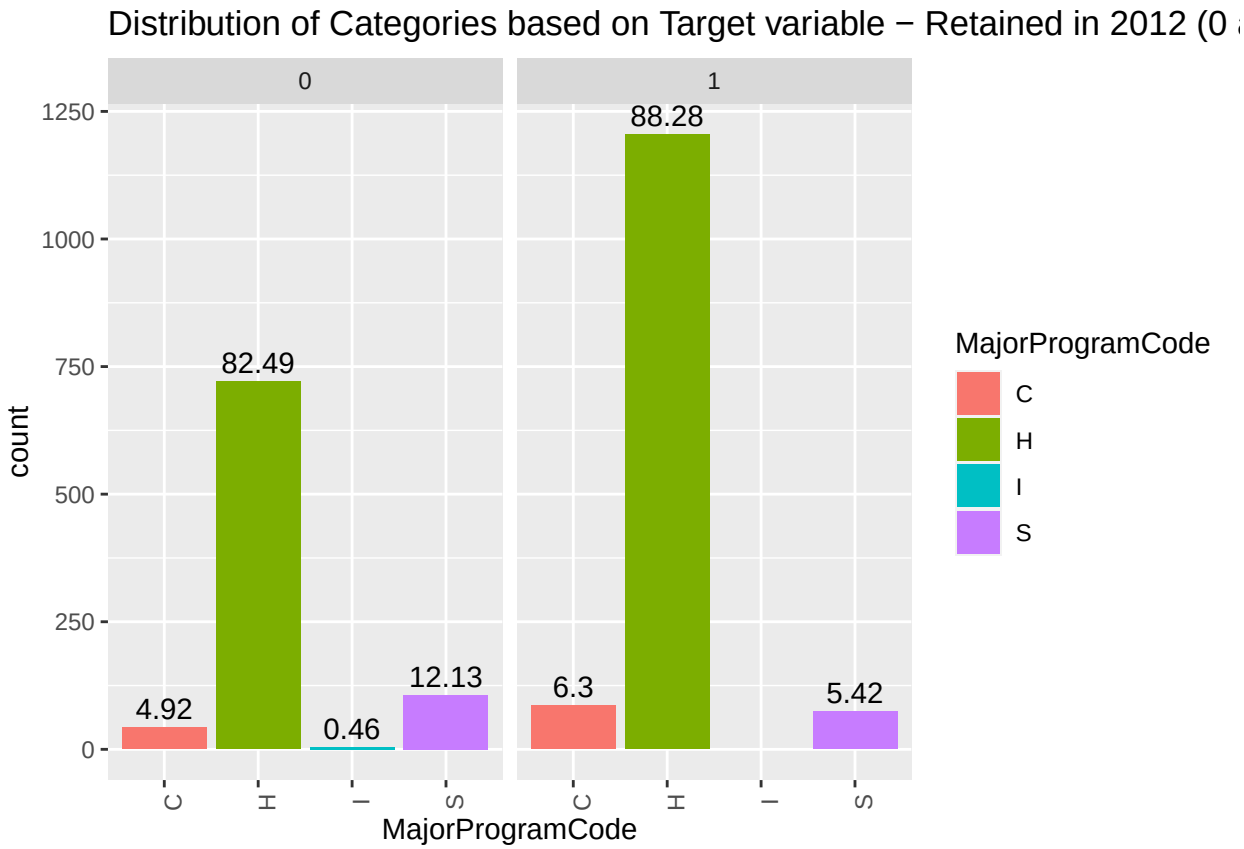

Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
1000
750
500
250
0
count
5.95
9.73
4.23
80.09
7.25
11.21
7.91
73.63
Catholic
CHD
Private non−Christian
PUBLIC
School.Type
School.Type
Catholic
CHD
Private non−Christian
PUBLIC

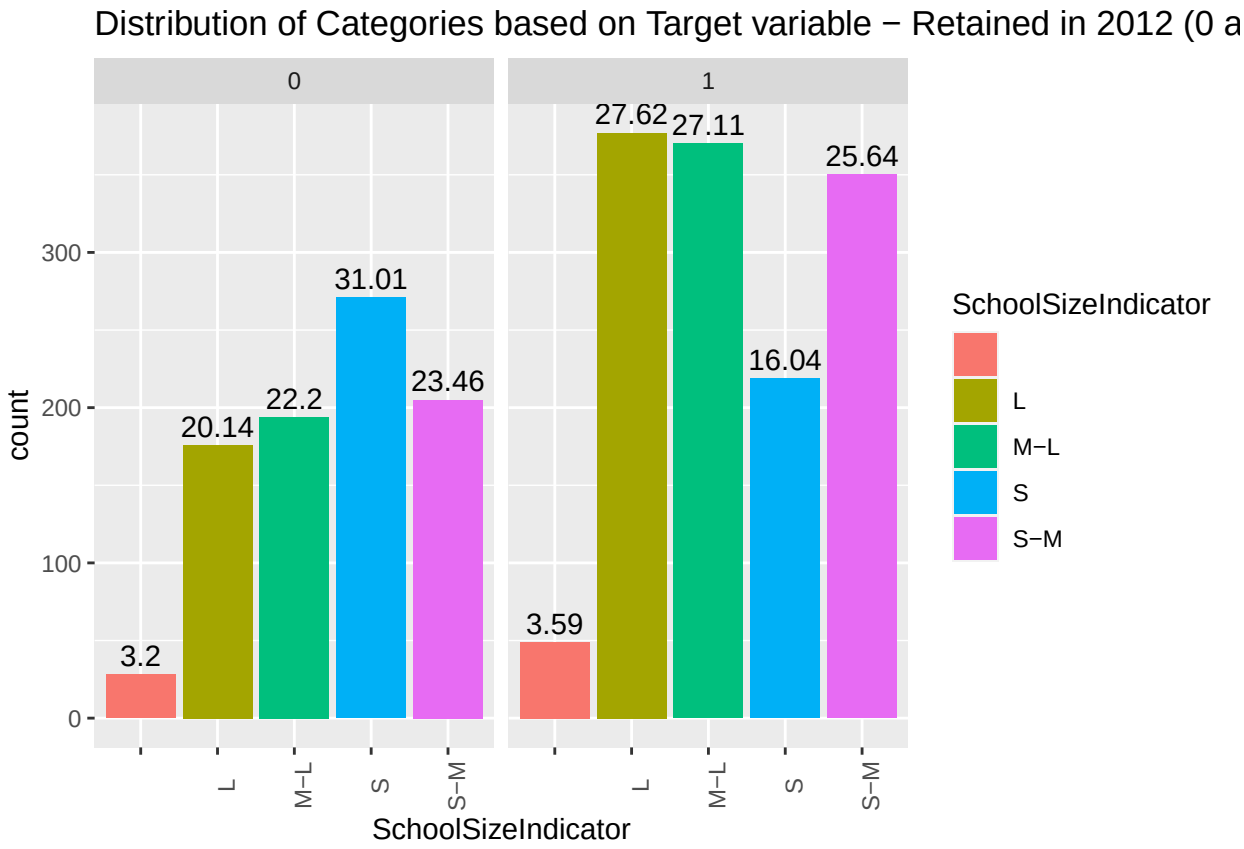

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
Income.Level
Income.Level
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
P1
P3
P4
P5
Q
Z

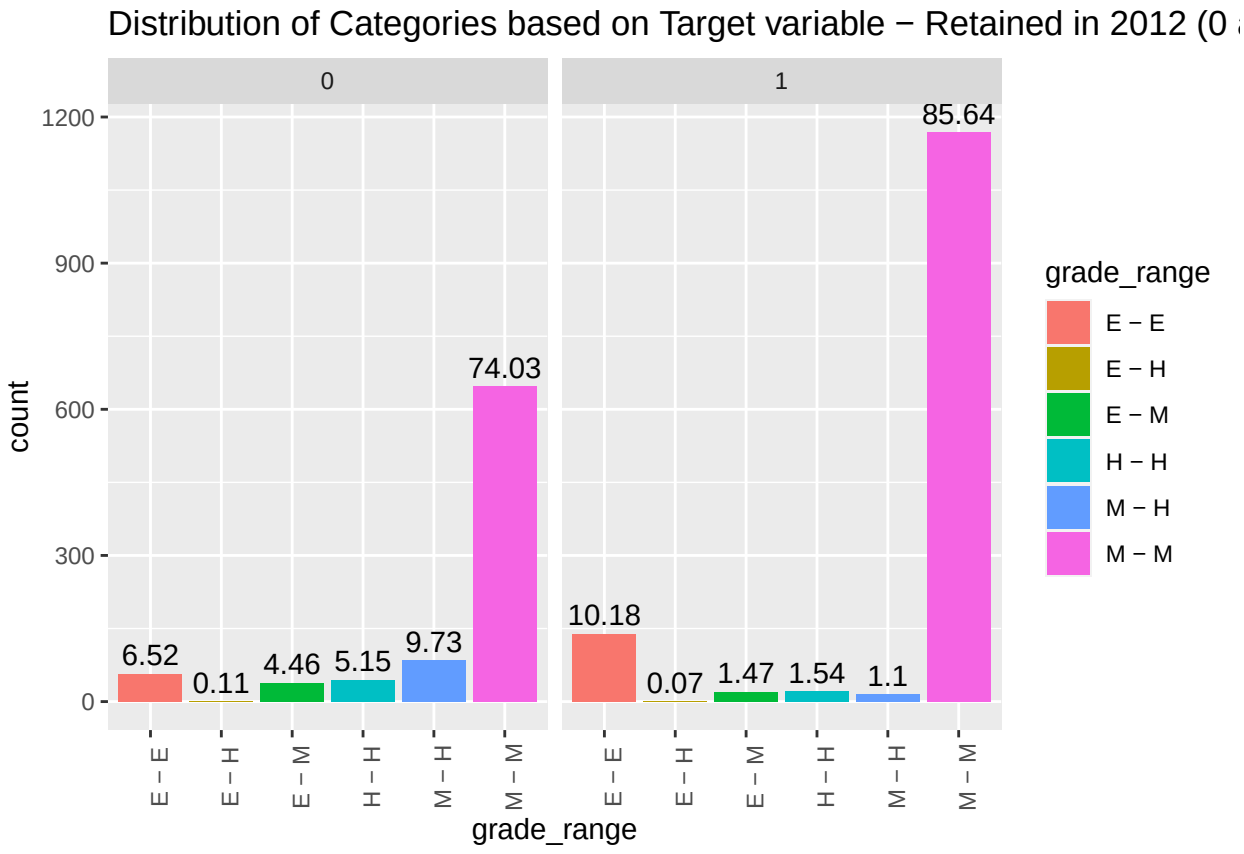

Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
count
1200
900
600
300
0
5.72
3.78
81.81
0.34
8.35
8.86
0.81
85.64
0.07
4.62
CA History
Costa Rica
East Coast
IL History
International
Science
SPR.Product.Type
SPR.Product.Type
CA History
Costa Rica
East Coast
IL History
International
Science

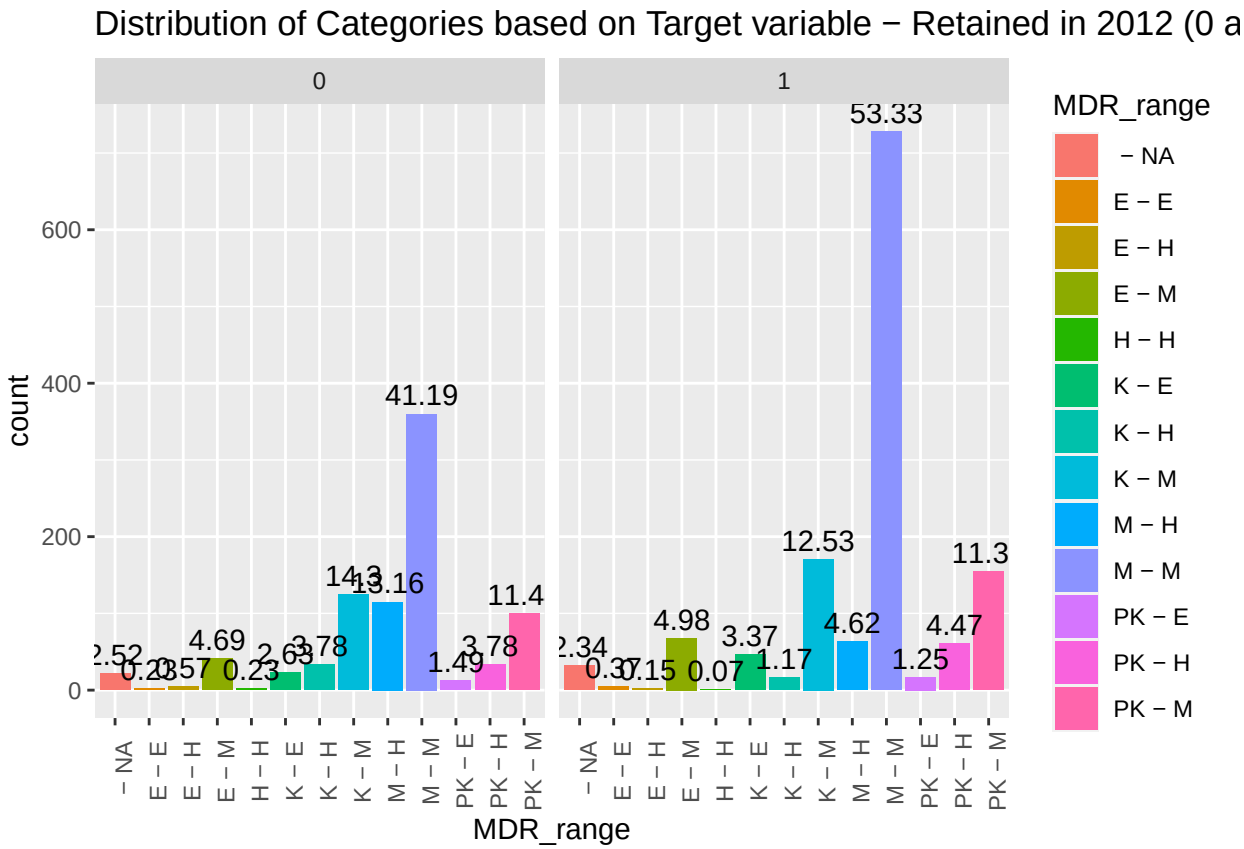

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
82.56
900
600
300
0
count
47.6
52.4
17.44
EXISTING
NEW
EXISTING
NEW
SPR.New.Existing
SPR.New.Existing
EXISTING
NEW

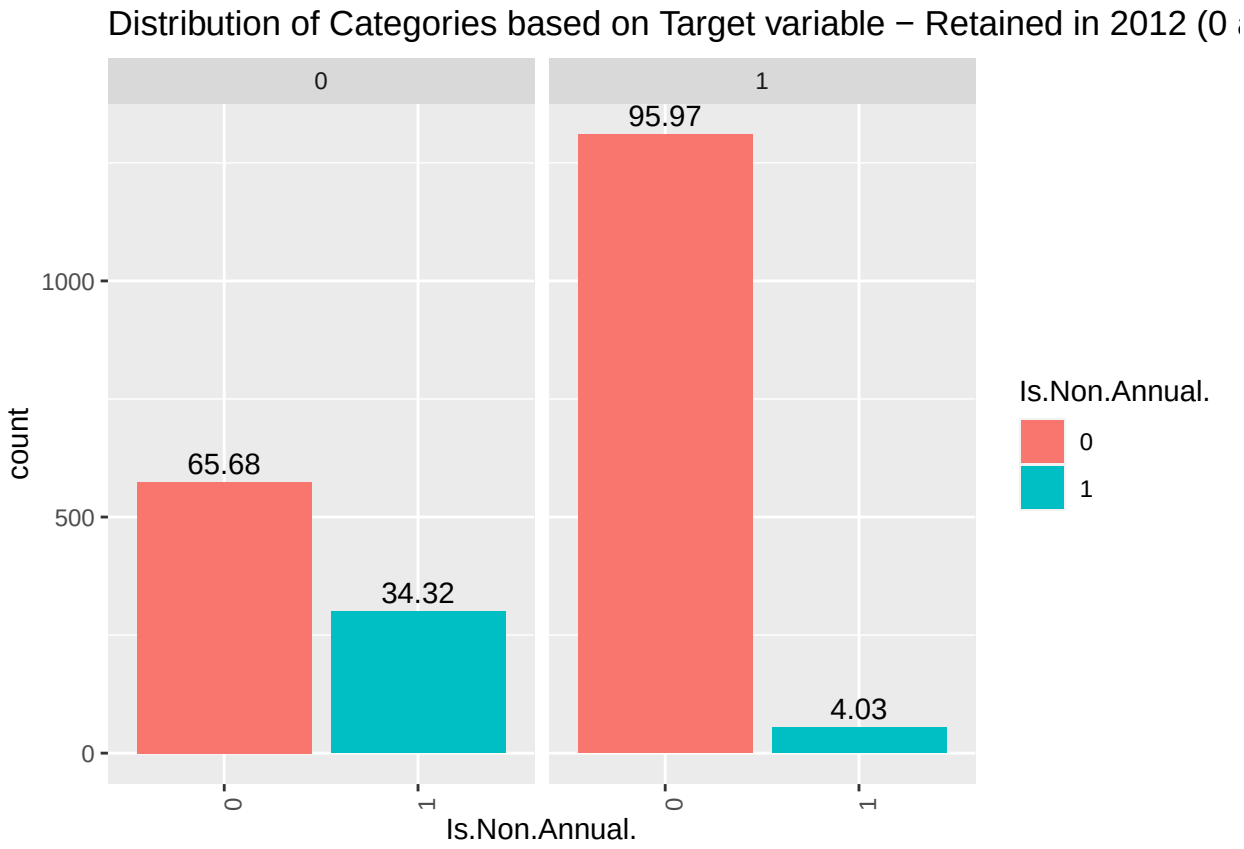

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
400
300
200
100
0
21.4
0.23
0.46
47.25
14.87
15.79
23.66
3.15
0.37
31.14
17.51
24.18
April
February
January
June
March
May
DepartureMonth
DepartureMonth
April
February
January
June
March
May

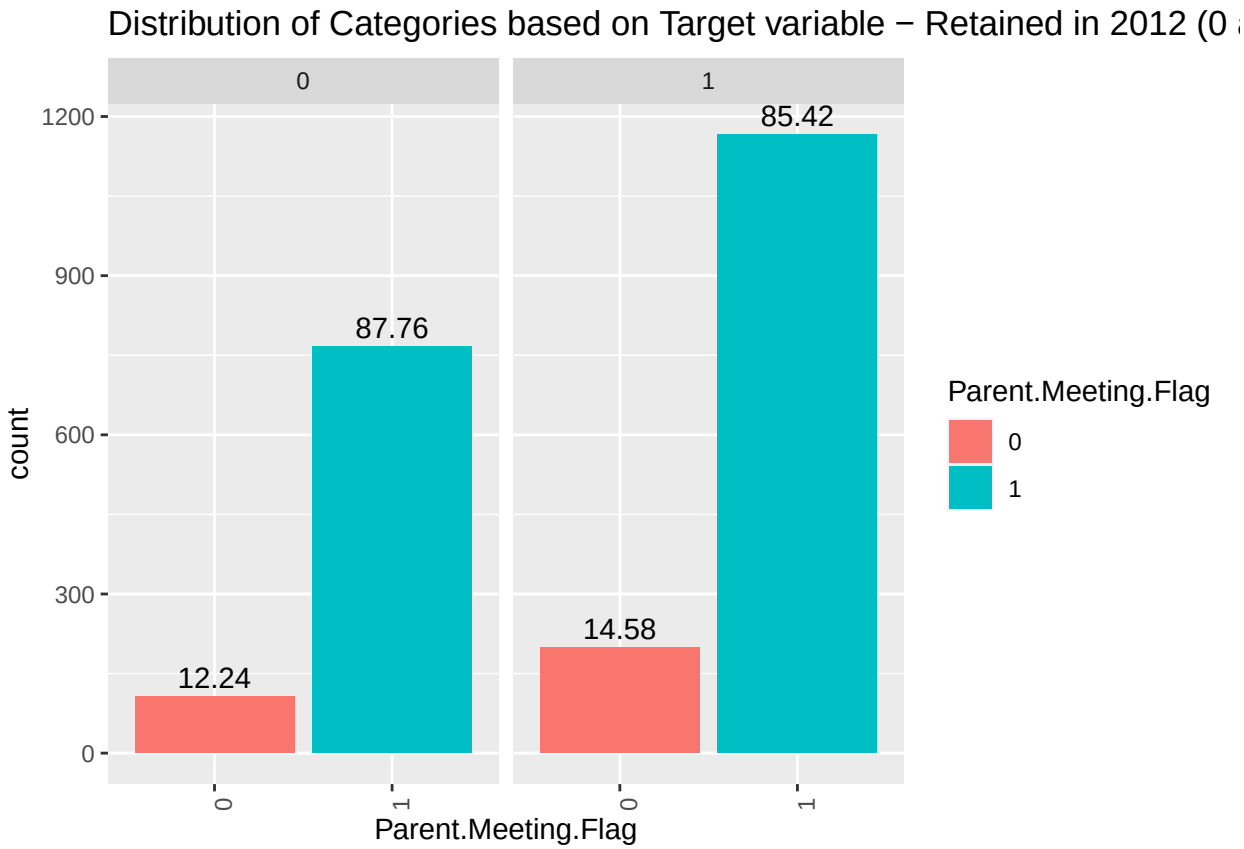

Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
1250
1000
750
500
250
0
count
82.49
4.92
0.46
12.13
88.28
6.3
5.42
C
H
I
S
C
H
I
S
MajorProgramCode
MajorProgramCode
C
H
I
S

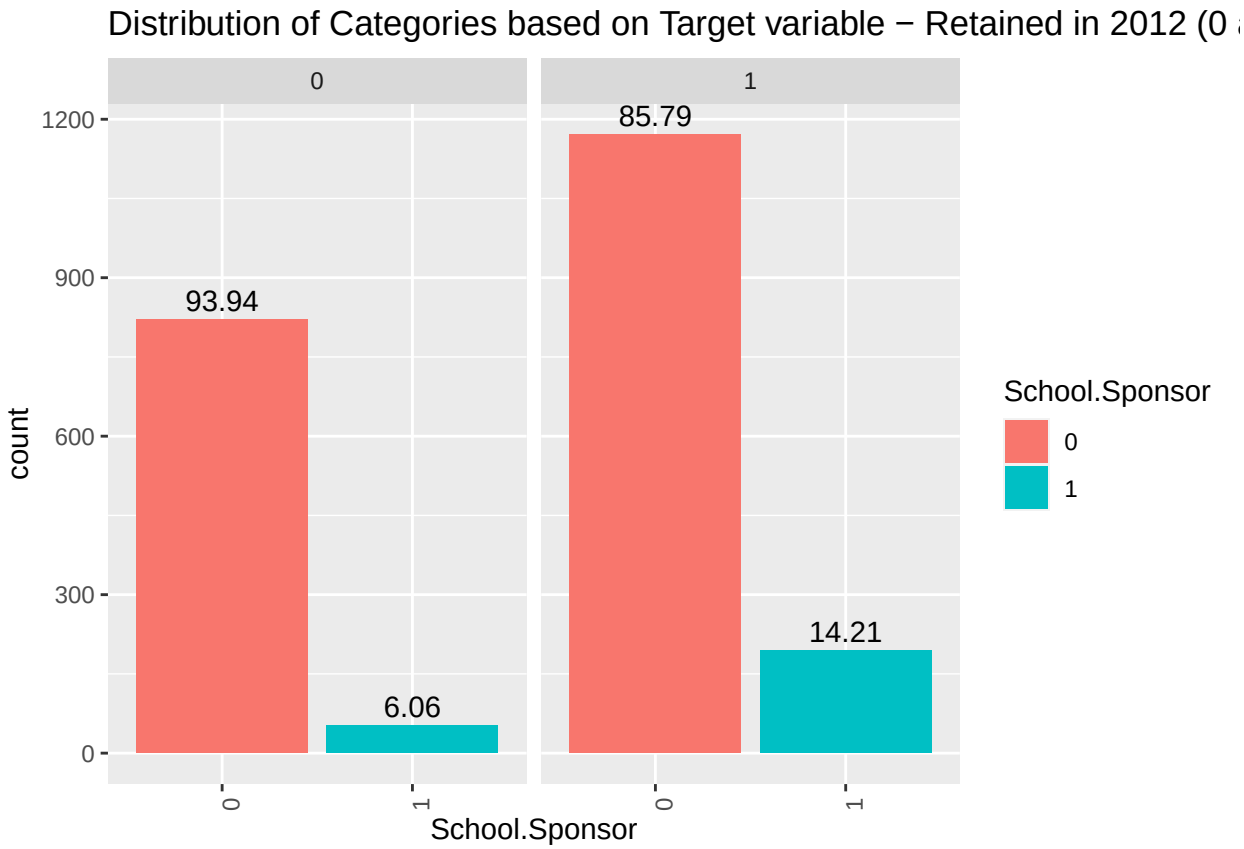

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
300
200
100
0
3.2
20.14
22.2
31.01
23.46
3.59
27.62
27.11
16.04
25.64
L
M−L
S
S−M
SchoolSizeIndicator
SchoolSizeIndicator
L
M−L
S
S−M

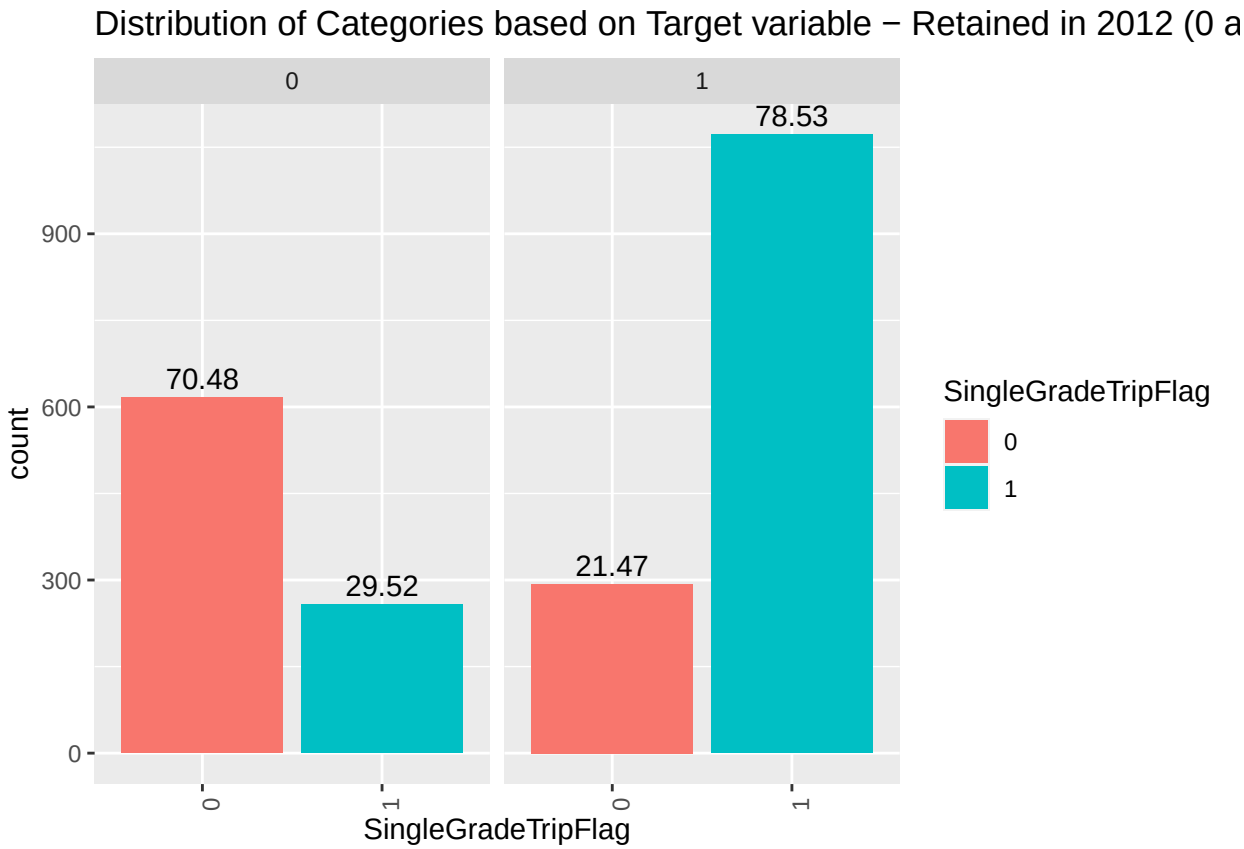

Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
count
1200
900
600
300
0
6.52
0.11
4.46
5.15
9.73
74.03
10.18
0.07
1.47
1.54
1.1
85.64
E − E
E − H
E − M
H − H
M − H
M − M
grade_range
grade_range
E − E
E − H
E − M
H − H
M − H
M − M

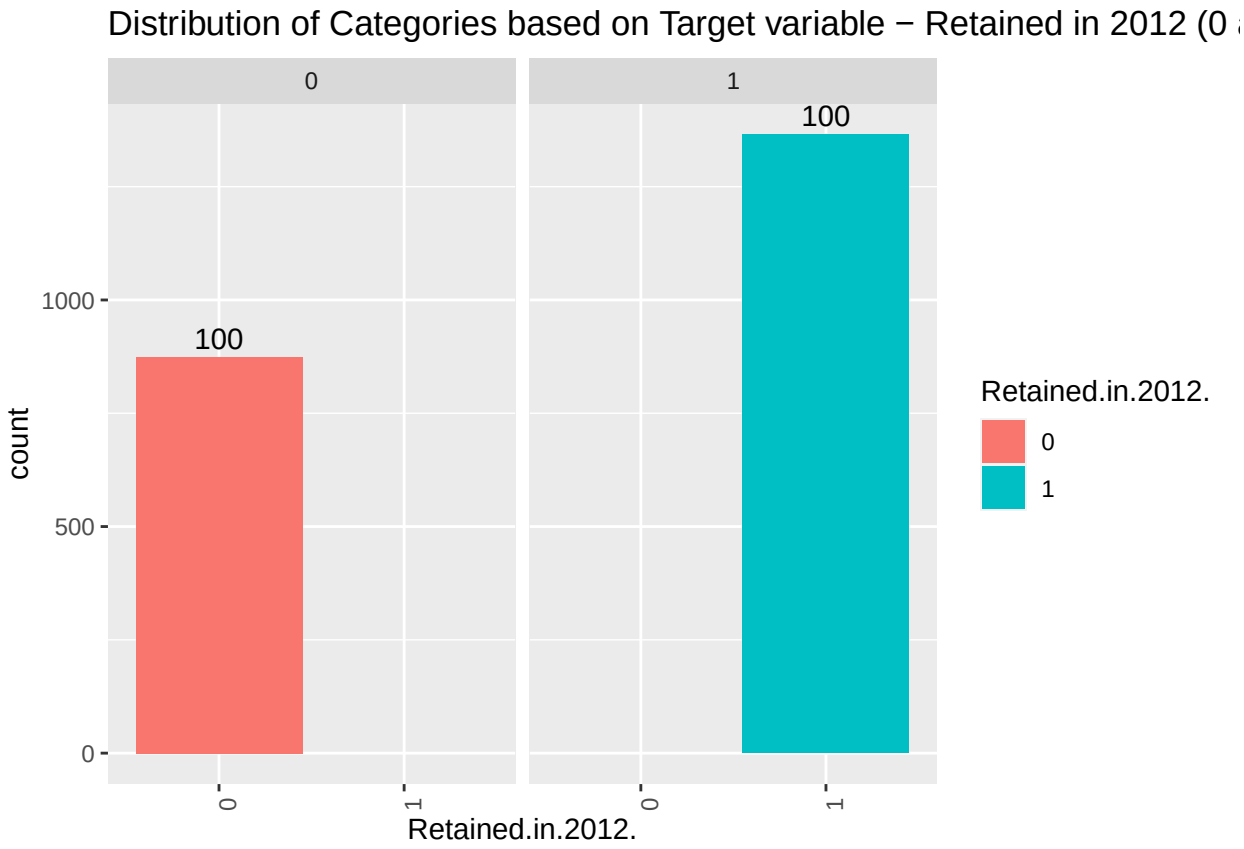

Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
600
400
200
0
41.19
14.3
13.16
11.4
4.69
3.78
1.49
53.33
12.53
11.3
4.98
4.62
4.47
3.37
1.17
1.25
0.07
− NA
E − E
E − H
E − M
H − H
K − E
K − H
K − M
M − H
M − M
PK − E
PK − H
PK − M
MDR_range

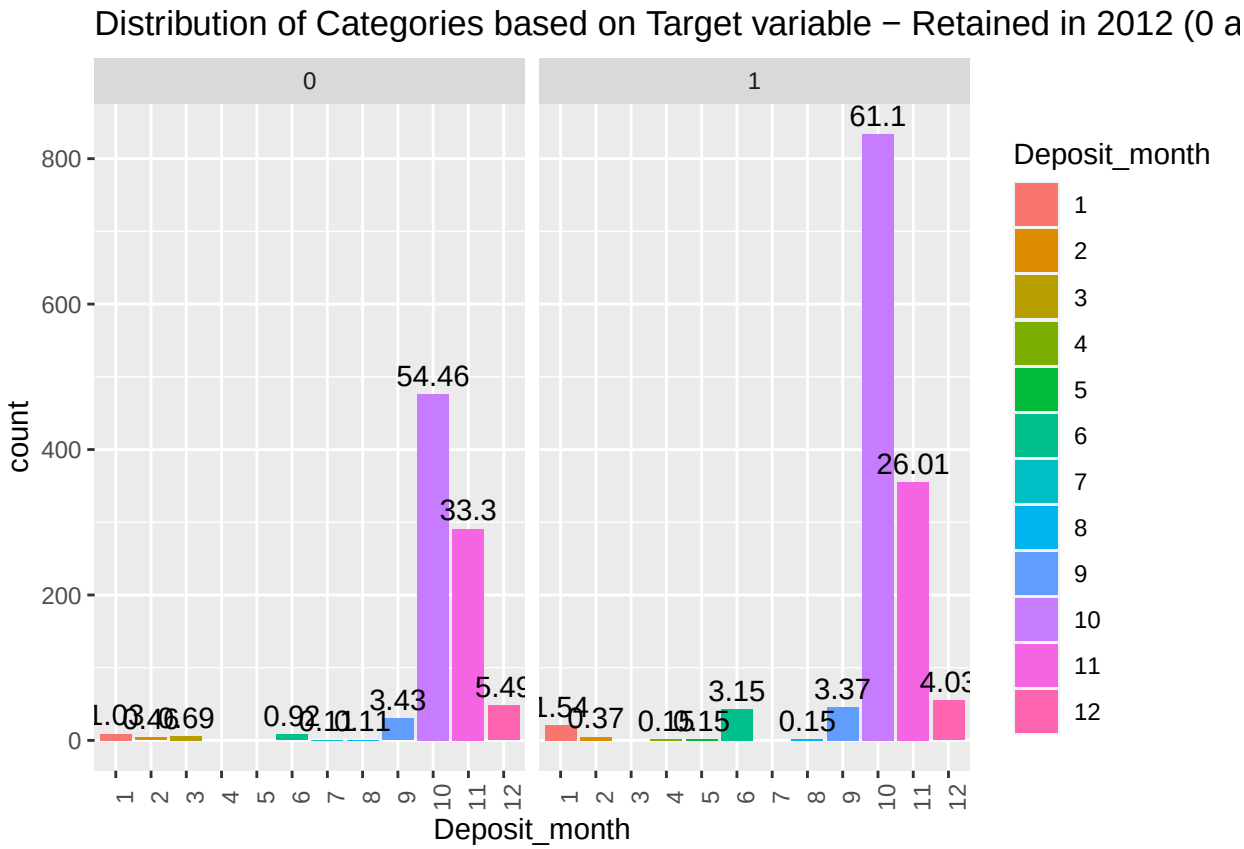

Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
95.97
65.68
34.32
4.03
1000
500
0
count
0
1
0
1
Is.Non.Annual.
Is.Non.Annual.
0
1

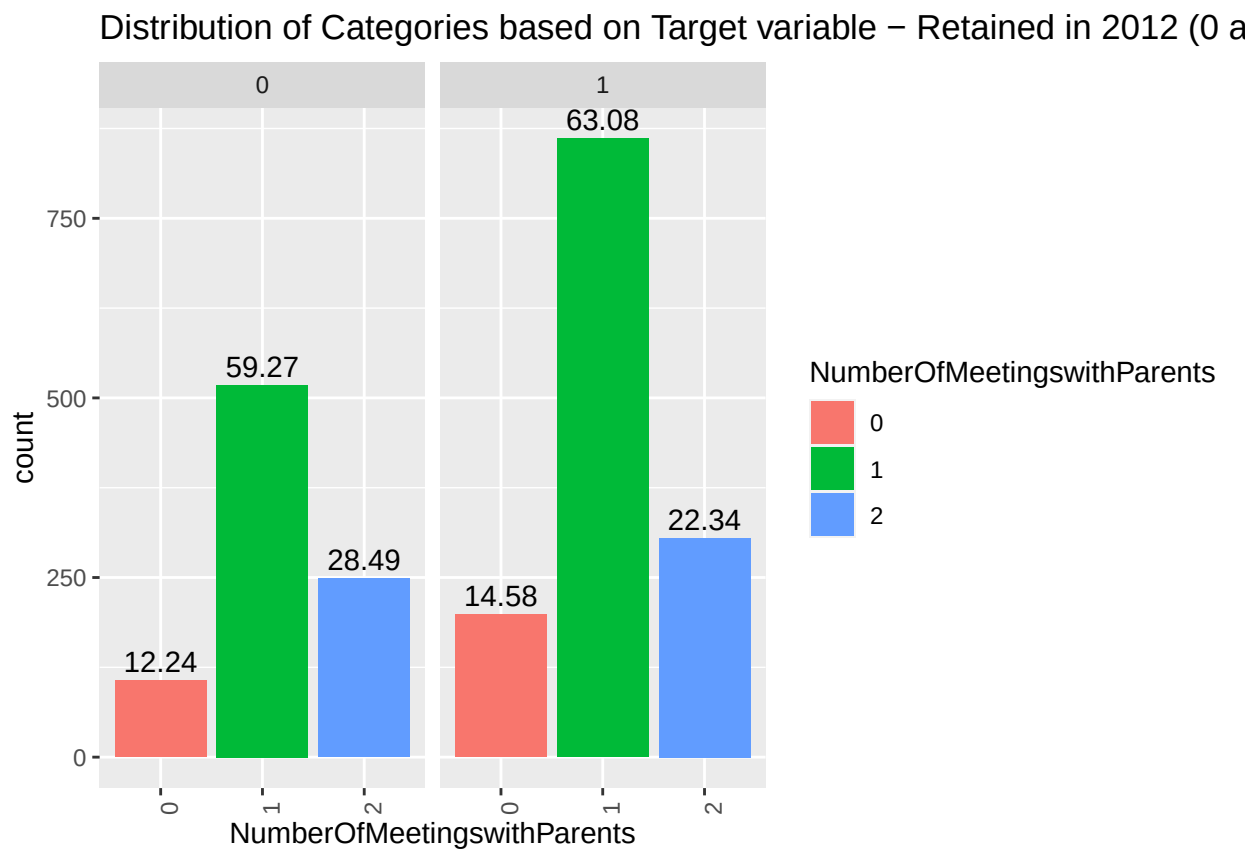

Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
1200
900
600
300
0
count
87.76
12.24
85.42
14.58
0
1
0
1
Parent.Meeting.Flag
Parent.Meeting.Flag
0
1


Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
1200
900
600
300
0
count
93.94
6.06
85.79
14.21
0
1
0
1
School.Sponsor
School.Sponsor
0
1


Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
78.53
900
70.48
600
count
300
29.52
21.47
0
0
1
0
1
SingleGradeTripFlag
SingleGradeTripFlag
0
1


Distribution of Categories based on Target variable − Retained in 2012 (0
0
1
100
100
1000
500
0
count
0
1
0
1
Retained.in.2012.
Retained.in.2012.
0
1


Distribution of Categories based on Target variable − Retained in 2012 (0 a
0
1
count
800
600
400
200
0
54.46
33.3
3.43
61.1
26.01
3.15
3.37
0.37
0.15
Deposit_month
1
2
3
4
5
6
7
8
9
10
11
12
Deposit_month




```
# Finding the proportion of Retention Succes and Failure in each categorical variable
for (i in cat){
  writeLines("\n\n")
  print("-------------------------------------------------------------------------------------------")
  print(paste("Input Variable :",i))
  subcat <- unique(dfn[i])
  print(paste("Categories in",i,":"))
  for (j in subcat){
    print(j)
  for (k in j){
    dfx <- filter(dfn, dfn[i]==k)
    t_cnt <- count(dfx)
    df_1 <- filter(df1, df1[i]==k)
    c1 <- count(df_1)
    df_0 <- filter(df0, df0[i]==k)
    c0 <- count(df_0)
    writeLines("\n")
    print(paste("Percentage(%) of RETENTION = 1 (SUCCESS) for category",k,"in variable",i,
                "is: ",round(c1/t_cnt*100,2),'%'))
    print(paste("Percentage(%) of RETENTION = 0 (FAIL) for category",k,"in variable",i,
                "is: ",round(c0/t_cnt*100,2),'%'))
    }
  }
}
```

```
##
```

```
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Program.Code"
## [1] "Categories in Program.Code :"
## [1] HS     HC     HD     HN     Others CD     HVP    SK     SC
## Levels: CD HC HD HN HS HVP Others SC SK
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category HS in variable Program.Code is:  67.44 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category HS in variable Program.Code is:  32.56 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category HC in variable Program.Code is:  64.09 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category HC in variable Program.Code is:  35.91 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category HD in variable Program.Code is:  62.78 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category HD in variable Program.Code is:  37.22 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category HN in variable Program.Code is:  45.45 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category HN in variable Program.Code is:  54.55 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Others in variable Program.Code is:  60.6
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Others in variable Program.Code is:  39.39 %
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category CD in variable Program.Code is:  67.59 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category CD in variable Program.Code is:  32.41 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category HVP in variable Program.Code is:  48.98 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category HVP in variable Program.Code is:  51.02 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category SK in variable Program.Code is:  36.92 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category SK in variable Program.Code is:  63.08 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category SC in variable Program.Code is:  25 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category SC in variable Program.Code is:  75 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Travel.Type"
## [1] "Categories in Travel.Type :"
## [1] A B T N
## Levels: A B N T
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category A in variable Travel.Type is:  59.38 %"
```

```
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category A in variable Travel.Type is:  40.62 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category B in variable Travel.Type is:  70.59 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category B in variable Travel.Type is:  29.41 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category T in variable Travel.Type is:  66.67 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category T in variable Travel.Type is:  33.33 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category N in variable Travel.Type is:  0 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category N in variable Travel.Type is:  100 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Poverty.Code"
## [1] "Categories in Poverty.Code :"
## [1] B C   D A E 0
## Levels:  0 A B C D E
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category B in variable Poverty.Code is:  58.98 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category B in variable Poverty.Code is:  41.02 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category C in variable Poverty.Code is:  52.75 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category C in variable Poverty.Code is:  47.25 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category  in variable Poverty.Code is:  67.26 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category  in variable Poverty.Code is:  32.74 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category D in variable Poverty.Code is:  42.42 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category D in variable Poverty.Code is:  57.58 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category A in variable Poverty.Code is:  73.95 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category A in variable Poverty.Code is:  26.05 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E in variable Poverty.Code is:  42.86 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E in variable Poverty.Code is:  57.14 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable Poverty.Code is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable Poverty.Code is:  50 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Region"
```

```
## [1] "Categories in Region :"
## [1] Southern California Other               Houston
## [4] Northern California Dallas              Pacific Northwest
## 6 Levels: Dallas Houston Northern California Other ... Southern California
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Southern California in variable Region is
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Southern California in variable Region is:  3
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Other in variable Region is:  61.34 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Other in variable Region is:  38.66 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Houston in variable Region is:  56.93 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Houston in variable Region is:  43.07 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Northern California in variable Region is
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Northern California in variable Region is:  3
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Dallas in variable Region is:  57.59 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Dallas in variable Region is:  42.41 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Pacific Northwest in variable Region is:
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Pacific Northwest in variable Region is:  56
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : School.Type"
## [1] "Categories in School.Type :"
## [1] PUBLIC                CHD                    Catholic
## [4] Private non-Christian
## Levels: Catholic CHD Private non-Christian PUBLIC
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category PUBLIC in variable School.Type is:  58.94
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category PUBLIC in variable School.Type is:  41.06 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category CHD in variable School.Type is:  64.29 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category CHD in variable School.Type is:  35.71 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Catholic in variable School.Type is:  65.5
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Catholic in variable School.Type is:  34.44 %
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Private non-Christian in variable School.T
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Private non-Christian in variable School.Type
##
```

```
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Income.Level"
## [1] "Categories in Income.Level :"
##  [1] Q  A  O     C  I  G  K  L  P5 N  H  E  M  P  P1 J  P3 Z  B  F  D  P4
## Levels:  A B C D E F G H I J K L M N O P P1 P3 P4 P5 Q Z
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Q in variable Income.Level is:  71.86 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Q in variable Income.Level is:  28.14 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category A in variable Income.Level is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category A in variable Income.Level is:  50 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category O in variable Income.Level is:  66.4 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category O in variable Income.Level is:  33.6 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category  in variable Income.Level is:  57.14 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category  in variable Income.Level is:  42.86 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category C in variable Income.Level is:  43.55 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category C in variable Income.Level is:  56.45 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category I in variable Income.Level is:  61.38 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category I in variable Income.Level is:  38.62 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category G in variable Income.Level is:  48.86 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category G in variable Income.Level is:  51.14 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category K in variable Income.Level is:  57.76 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category K in variable Income.Level is:  42.24 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category L in variable Income.Level is:  60.78 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category L in variable Income.Level is:  39.22 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category P5 in variable Income.Level is:  100 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category P5 in variable Income.Level is:  0 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category N in variable Income.Level is:  58.57 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category N in variable Income.Level is:  41.43 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category H in variable Income.Level is:  52.5 %"
```

```
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category H in variable Income.Level is:  47.5 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E in variable Income.Level is:  43.42 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E in variable Income.Level is:  56.58 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category M in variable Income.Level is:  67.58 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category M in variable Income.Level is:  32.42 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category P in variable Income.Level is:  79.9 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category P in variable Income.Level is:  20.1 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category P1 in variable Income.Level is:  100 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category P1 in variable Income.Level is:  0 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category J in variable Income.Level is:  50.33 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category J in variable Income.Level is:  49.67 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category P3 in variable Income.Level is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category P3 in variable Income.Level is:  50 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Z in variable Income.Level is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Z in variable Income.Level is:  50 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category B in variable Income.Level is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category B in variable Income.Level is:  50 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category F in variable Income.Level is:  48.98 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category F in variable Income.Level is:  51.02 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category D in variable Income.Level is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category D in variable Income.Level is:  50 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category P4 in variable Income.Level is:  100 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category P4 in variable Income.Level is:  0 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : SPR.Product.Type"
## [1] "Categories in SPR.Product.Type :"
## [1] CA History    East Coast    Science       Costa Rica    IL History
## [6] International
## 6 Levels: CA History Costa Rica East Coast IL History ... Science
```

```
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category CA History in variable SPR.Product.Type is
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category CA History in variable SPR.Product.Type is:
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category East Coast in variable SPR.Product.Type is
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category East Coast in variable SPR.Product.Type is:
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Science in variable SPR.Product.Type is:
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Science in variable SPR.Product.Type is:  53
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category Costa Rica in variable SPR.Product.Type is
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category Costa Rica in variable SPR.Product.Type is:
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category IL History in variable SPR.Product.Type is
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category IL History in variable SPR.Product.Type is:
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category International in variable SPR.Product.Type
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category International in variable SPR.Product.Type is
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : SPR.New.Existing"
## [1] "Categories in SPR.New.Existing :"
## [1] EXISTING NEW
## Levels: EXISTING NEW
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category EXISTING in variable SPR.New.Existing is:
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category EXISTING in variable SPR.New.Existing is:  20
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category NEW in variable SPR.New.Existing is:  34.1
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category NEW in variable SPR.New.Existing is:  65.8 %
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : DepartureMonth"
## [1] "Categories in DepartureMonth :"
## [1] January  February March    April    May      June
## Levels: April February January June March May
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category January in variable DepartureMonth is:  55
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category January in variable DepartureMonth is:  44.44
##
##
```

```
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category February in variable DepartureMonth is:  9
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category February in variable DepartureMonth is:  4.44
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category March in variable DepartureMonth is:  64.7
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category March in variable DepartureMonth is:  35.23 %
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category April in variable DepartureMonth is:  63.3
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category April in variable DepartureMonth is:  36.67 %
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category May in variable DepartureMonth is:  70.51
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category May in variable DepartureMonth is:  29.49 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category June in variable DepartureMonth is:  50.72
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category June in variable DepartureMonth is:  49.28 %
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : MajorProgramCode"
## [1] "Categories in MajorProgramCode :"
## [1] H S I C
## Levels: C H I S
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category H in variable MajorProgramCode is:  62.56
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category H in variable MajorProgramCode is:  37.44 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category S in variable MajorProgramCode is:  41.11
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category S in variable MajorProgramCode is:  58.89 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category I in variable MajorProgramCode is:  0 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category I in variable MajorProgramCode is:  100 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category C in variable MajorProgramCode is:  66.67
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category C in variable MajorProgramCode is:  33.33 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : SchoolSizeIndicator"
## [1] "Categories in SchoolSizeIndicator :"
## [1] L       M-L S   S-M
## Levels:  L M-L S S-M
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category L in variable SchoolSizeIndicator is:  68
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category L in variable SchoolSizeIndicator is:  31.83
```

```
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category  in variable SchoolSizeIndicator is:  63.6
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category  in variable SchoolSizeIndicator is:  36.36 %
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category M-L in variable SchoolSizeIndicator is:  6
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category M-L in variable SchoolSizeIndicator is:  34.4
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category S in variable SchoolSizeIndicator is:  44
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category S in variable SchoolSizeIndicator is:  55.31
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category S-M in variable SchoolSizeIndicator is:  6
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category S-M in variable SchoolSizeIndicator is:  36.9
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : grade_range"
## [1] "Categories in grade_range :"
## [1] E - E M - M M - H H - H E - M E - H
## Levels: E - E E - H E - M H - H M - H M - M
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E - E in variable grade_range is:  70.92 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E - E in variable grade_range is:  29.08 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category M - M in variable grade_range is:  64.37 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category M - M in variable grade_range is:  35.63 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category M - H in variable grade_range is:  15 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category M - H in variable grade_range is:  85 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category H - H in variable grade_range is:  31.82 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category H - H in variable grade_range is:  68.18 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E - M in variable grade_range is:  33.9 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E - M in variable grade_range is:  66.1 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E - H in variable grade_range is:  50 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E - H in variable grade_range is:  50 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : MDR_range"
## [1] "Categories in MDR_range :"
```

```
##  [1] K - E  M - M   - NA  H - H  M - H  PK - M K - H  E - M  PK - H K - M
## [11] E - E  PK - E E - H
## 13 Levels:  - NA E - E E - H E - M H - H K - E K - H K - M M - H ... PK - M
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category K - E in variable MDR_range is:  66.67 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category K - E in variable MDR_range is:  33.33 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category M - M in variable MDR_range is:  66.91 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category M - M in variable MDR_range is:  33.09 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category  - NA in variable MDR_range is:  59.26 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category  - NA in variable MDR_range is:  40.74 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category H - H in variable MDR_range is:  33.33 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category H - H in variable MDR_range is:  66.67 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category M - H in variable MDR_range is:  35.39 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category M - H in variable MDR_range is:  64.61 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category PK - M in variable MDR_range is:  60.78 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category PK - M in variable MDR_range is:  39.22 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category K - H in variable MDR_range is:  32.65 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category K - H in variable MDR_range is:  67.35 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E - M in variable MDR_range is:  62.39 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E - M in variable MDR_range is:  37.61 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category PK - H in variable MDR_range is:  64.89 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category PK - H in variable MDR_range is:  35.11 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category K - M in variable MDR_range is:  57.77 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category K - M in variable MDR_range is:  42.23 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E - E in variable MDR_range is:  71.43 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E - E in variable MDR_range is:  28.57 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category PK - E in variable MDR_range is:  56.67 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category PK - E in variable MDR_range is:  43.33 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category E - H in variable MDR_range is:  28.57 %"
```

```
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category E - H in variable MDR_range is:  71.43 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Is.Non.Annual."
## [1] "Categories in Is.Non.Annual. :"
## [1] 0 1
## Levels: 0 1
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable Is.Non.Annual. is:  69.53 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable Is.Non.Annual. is:  30.47 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable Is.Non.Annual. is:  15.49 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable Is.Non.Annual. is:  84.51 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Parent.Meeting.Flag"
## [1] "Categories in Parent.Meeting.Flag :"
## [1] 1 0
## Levels: 0 1
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable Parent.Meeting.Flag is:  60
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable Parent.Meeting.Flag is:  39.68
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable Parent.Meeting.Flag is:  65
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable Parent.Meeting.Flag is:  34.97
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : School.Sponsor"
## [1] "Categories in School.Sponsor :"
## [1] 1 0
## Levels: 0 1
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable School.Sponsor is:  78.54 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable School.Sponsor is:  21.46 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable School.Sponsor is:  58.79 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable School.Sponsor is:  41.21 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : SingleGradeTripFlag"
```

```
## [1] "Categories in SingleGradeTripFlag :"
## [1] 1 0
## Levels: 0 1
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable SingleGradeTripFlag is:  80
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable SingleGradeTripFlag is:  19.4 %
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable SingleGradeTripFlag is:  32
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable SingleGradeTripFlag is:  67.77
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Retained.in.2012."
## [1] "Categories in Retained.in.2012. :"
## [1] 1 0
## Levels: 0 1
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable Retained.in.2012. is:  100 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable Retained.in.2012. is:  0 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable Retained.in.2012. is:  0 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable Retained.in.2012. is:  100 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : Deposit_month"
## [1] "Categories in Deposit_month :"
##  [1] 8  11 10 1  9  6  12 2  4  7  5  3
## Levels: 1 2 3 4 5 6 7 8 9 10 11 12
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 8 in variable Deposit_month is:  66.67 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 8 in variable Deposit_month is:  33.33 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 11 in variable Deposit_month is:  54.95 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 11 in variable Deposit_month is:  45.05 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 10 in variable Deposit_month is:  63.66 %
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 10 in variable Deposit_month is:  36.34 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable Deposit_month is:  70 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable Deposit_month is:  30 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 9 in variable Deposit_month is:  60.53 %"
```

```
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 9 in variable Deposit_month is:  39.47 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 6 in variable Deposit_month is:  84.31 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 6 in variable Deposit_month is:  15.69 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 12 in variable Deposit_month is:  53.4 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 12 in variable Deposit_month is:  46.6 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 2 in variable Deposit_month is:  55.56 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 2 in variable Deposit_month is:  44.44 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 4 in variable Deposit_month is:  100 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 4 in variable Deposit_month is:  0 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 7 in variable Deposit_month is:  0 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 7 in variable Deposit_month is:  100 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 5 in variable Deposit_month is:  100 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 5 in variable Deposit_month is:  0 %"
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 3 in variable Deposit_month is:  0 %"
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 3 in variable Deposit_month is:  100 %"
##
##
##
## [1] "-----------------------------------------------------------------------------------------"
## [1] "Input Variable : NumberOfMeetingswithParents"
## [1] "Categories in NumberOfMeetingswithParents :"
## [1] 1 2 0
## Levels: 0 1 2
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 1 in variable NumberOfMeetingswithParents
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 1 in variable NumberOfMeetingswithParents is
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 2 in variable NumberOfMeetingswithParents
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 2 in variable NumberOfMeetingswithParents is
##
##
## [1] "Percentage(%) of RETENTION = 1 (SUCCESS) for category 0 in variable NumberOfMeetingswithParents
## [1] "Percentage(%) of RETENTION = 0 (FAIL) for category 0 in variable NumberOfMeetingswithParents is
```

from the above plots and distribution we can conclude that the following variables have a significant impact on the Target variable: - SPR.New.Existing - grade_range - Is.Non.Annual - SingleGradeTripFlag - School.Sponsor

EXPLORATION OF NUMERICAL VARIABLES

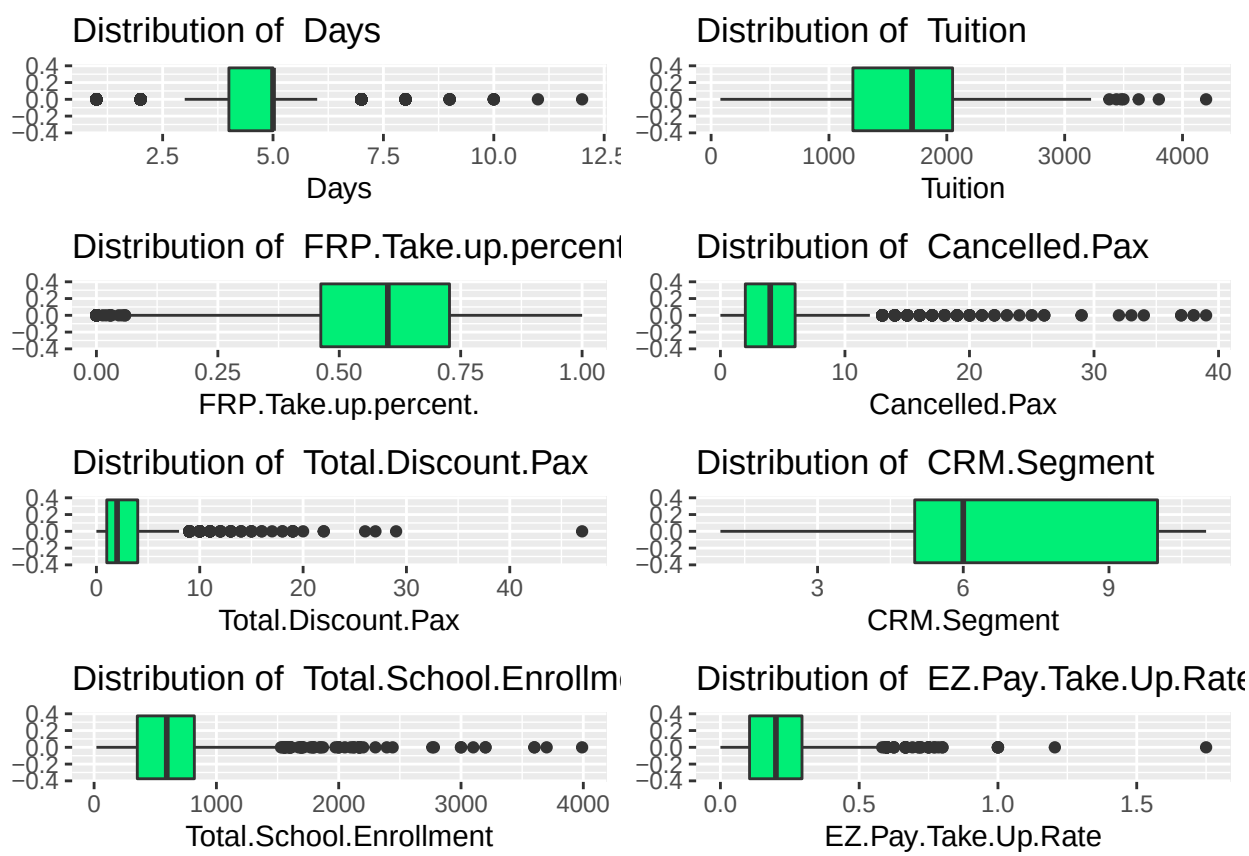
```
print("Numerical variables in the dataset :") #displaying numerical columns
```

```
## [1] "Numerical variables in the dataset :"
```

```
num
```

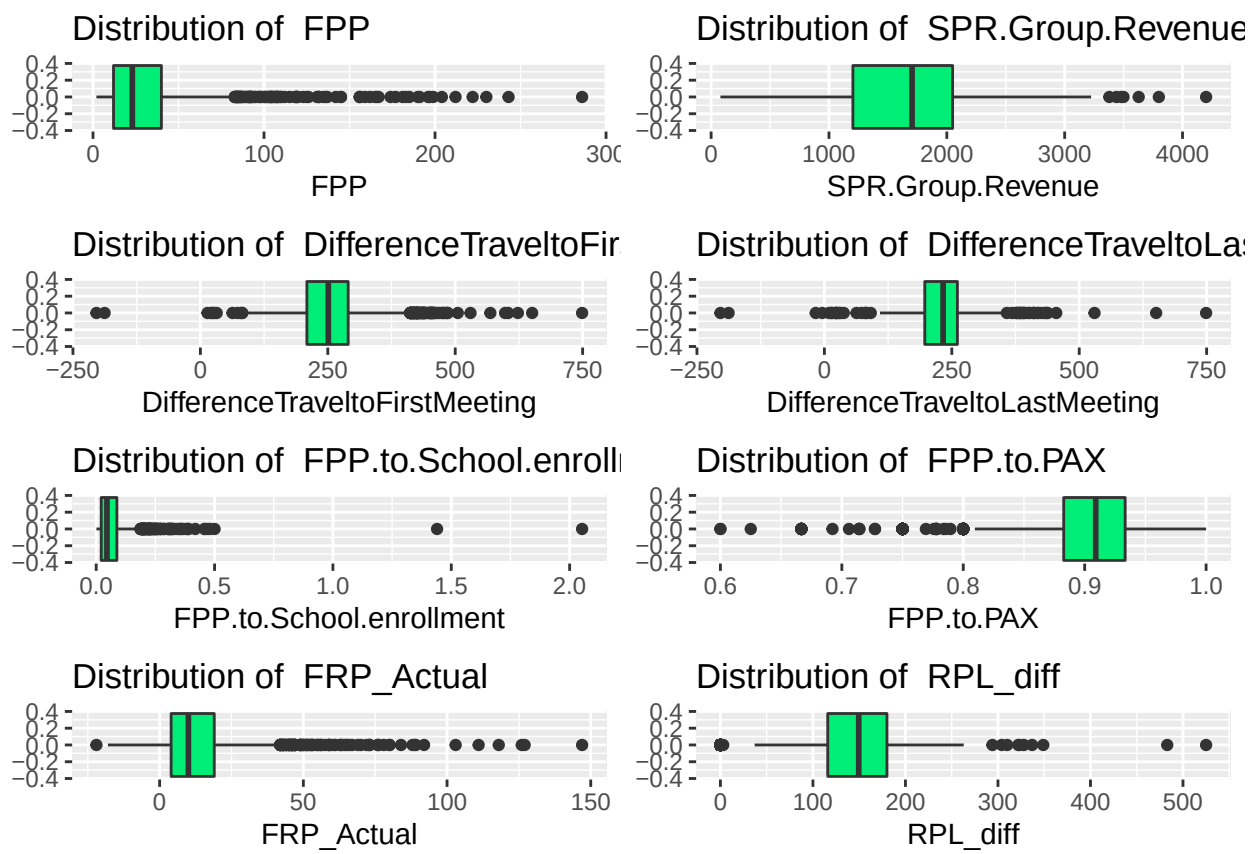
```
##  [1] "Days"                            "Tuition"
##  [3] "FRP.Take.up.percent."            "Cancelled.Pax"
##  [5] "Total.Discount.Pax"              "CRM.Segment"
##  [7] "Total.School.Enrollment"         "EZ.Pay.Take.Up.Rate"
##  [9] "FPP"                             "SPR.Group.Revenue"
## [11] "DifferenceTraveltoFirstMeeting" "DifferenceTraveltoLastMeeting"
## [13] "FPP.to.School.enrollment"        "FPP.to.PAX"
## [15] "FRP_Actual"                      "RPL_diff"
```

```
plot_list <- list()
n=1
for (i in num[1:8]){
  plot_list[[n]] <- ggplot(dfn,aes_string(x=i))+geom_boxplot(fill='springgreen2')+
    ggtitle(paste("Distribution of ",i))
  n=n+1
}
grid.arrange(grobs=plot_list,ncol=2)
```



```r
# SPlitting into 2 loops since output in markdown makes it hard to infer due to
# plot scale constrain

plot_list <- list()
n=1
for (i in num[9:16]){
  plot_list[[n]] <- ggplot(dfn,aes_string(x=i))+geom_boxplot(fill='springgreen2')+
    ggtitle(paste("Distribution of ",i))
  n=n+1
}
grid.arrange(grobs=plot_list,ncol=2)
```

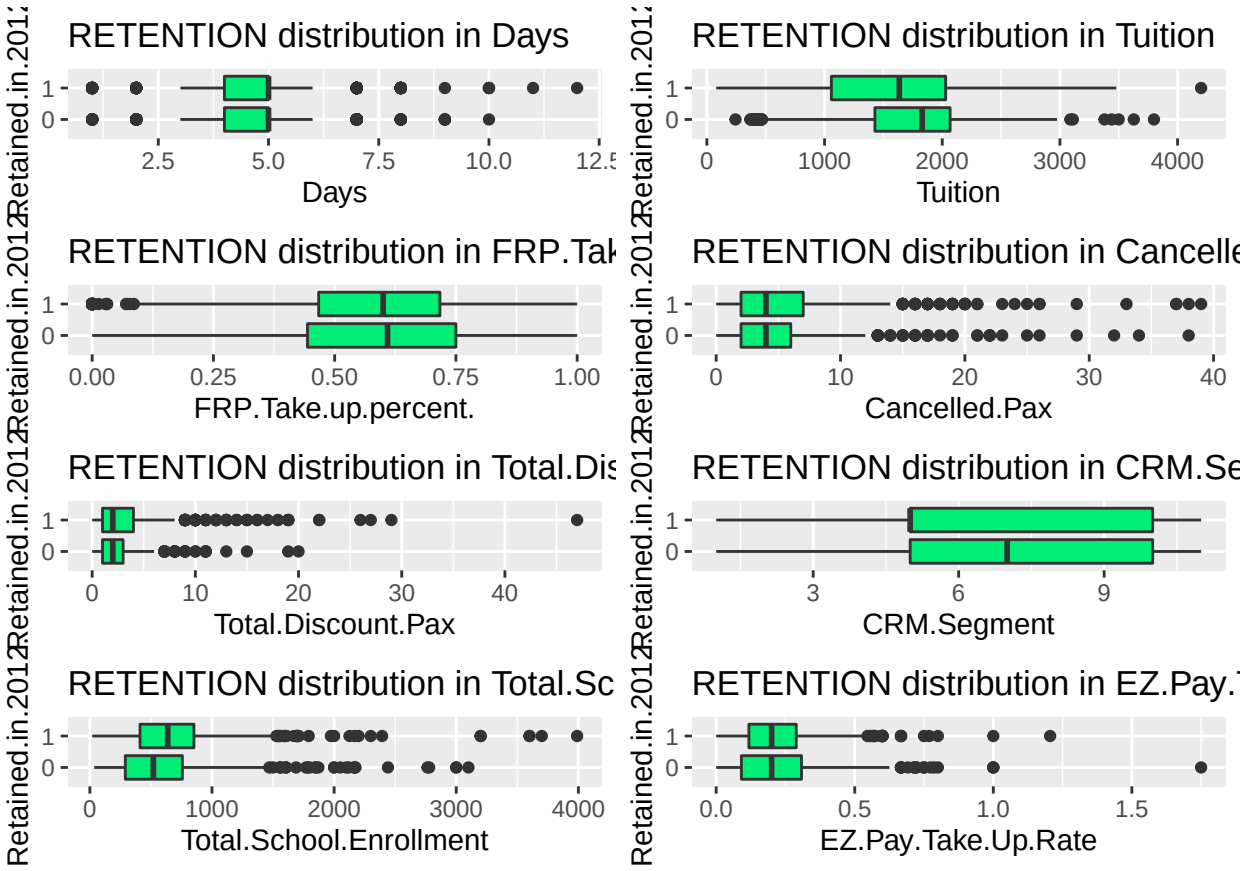
```
## Don't know how to automatically pick scale for object of type difftime. Defaulting to continuous.
```



NUMERICAL VARIABLES VS TARGET

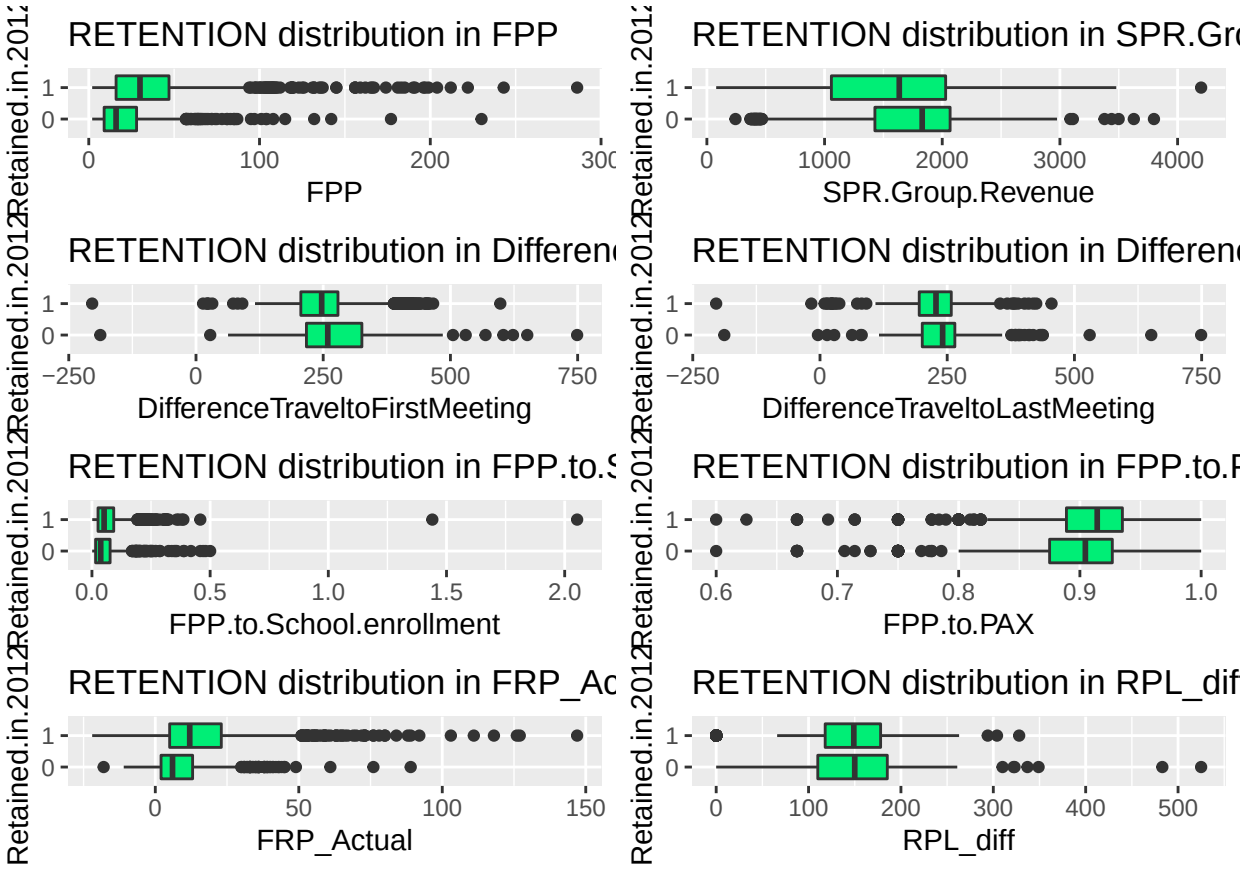
```r
# Plots to depict the distribution of Success/Failure of Retention in 2012
# with respect to numerical variables

plot_list <- list()
n=1
for (i in num[1:8]){
plot_list[[n]] <- ggplot(dfn,aes_string(x=i, y="Retained.in.2012.",
                                        group="Retained.in.2012."))+
geom_boxplot(fill="springgreen2")+
```

```
ggtitle(paste("RETENTION distribution in",i))
n=n+1
}
grid.arrange(grobs=plot_list,ncol=2)
```




```
plot_list <- list()
n=1
for (i in num[9:16]){
plot_list[[n]] <- ggplot(dfn,aes_string(x=i, y="Retained.in.2012.",
                                        group="Retained.in.2012."))+
geom_boxplot(fill="springgreen2")+
ggtitle(paste("RETENTION distribution in",i))
n=n+1
}
grid.arrange(grobs=plot_list,ncol=2)
```

```
## Don't know how to automatically pick scale for object of type difftime. Defaulting to continuous.
```

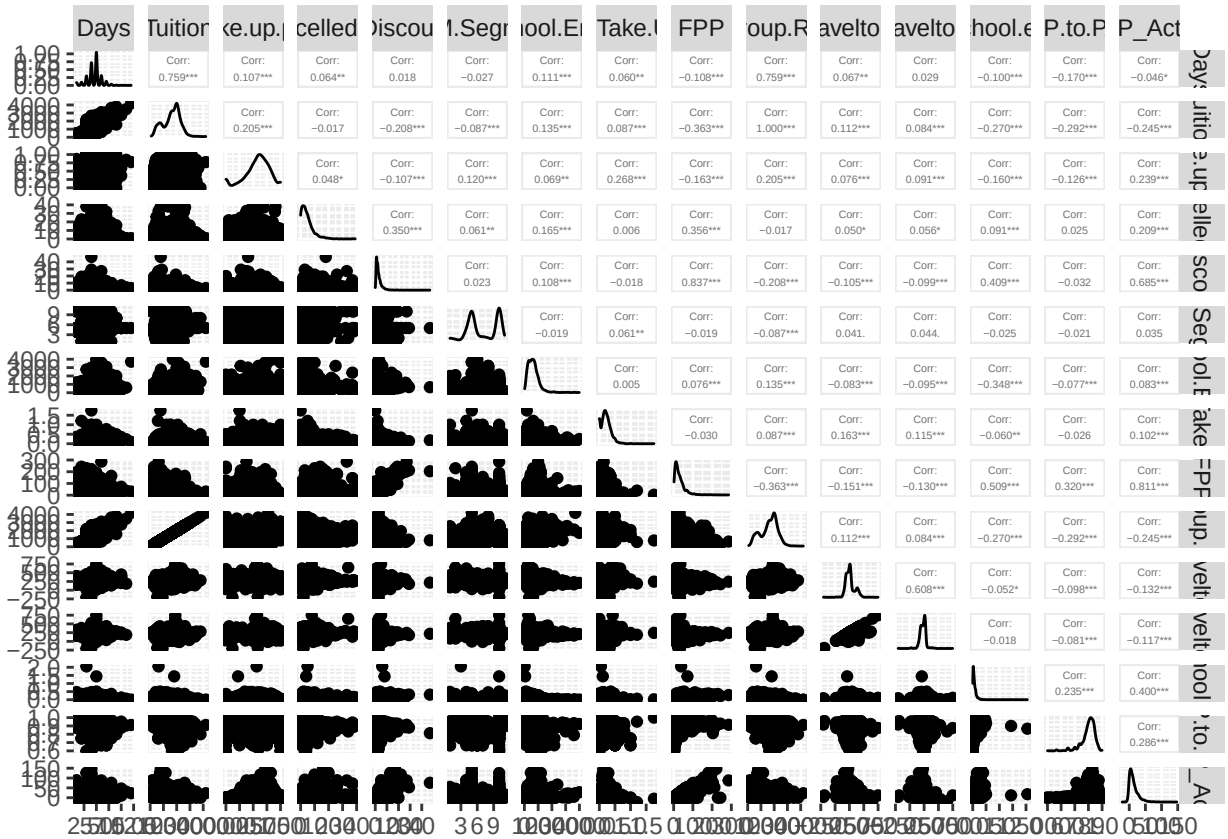


If there's a noticeable difference in the position of boxes for 1 and 0 in the above plots then we can say that the target has a visible dependency on that particular variable. Extending the above concept we can observe that the below numerical variables show an effect on target variable: - Tuition - Total.Discount.pax - FPP - FRP.Actuals - FPP.to.Pax

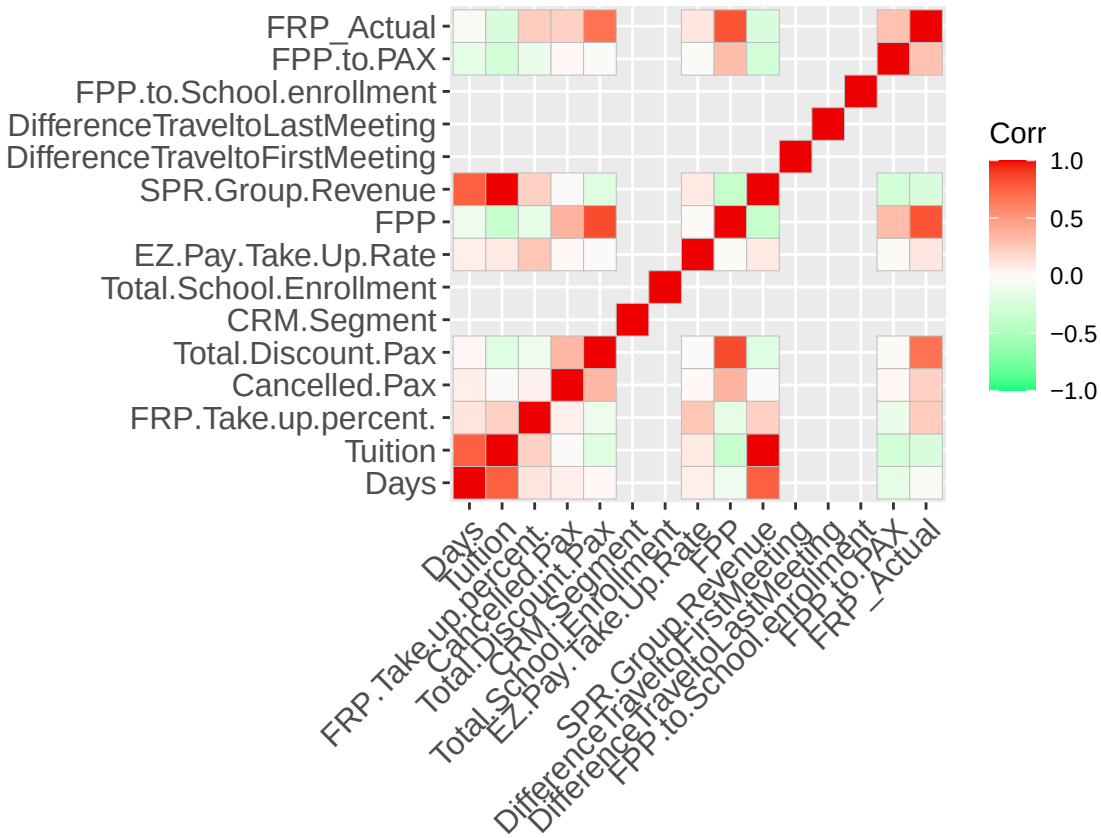
CORELATION BETWEEN NUMERIC VARIABLES

(checking MULTICOLLINEARITY)

```
# pair plot for input varibales
ggpairs(dfn[num[1:15]], upper = list(continuous = wrap("cor", size = 1.5)))
```



```
ggcorrplot(cor(dfn[num[1:15]]), ggtheme = 'theme_dark', show.legend = TRUE, colors=c('springgreen1','sn
```



While treating multicollinearity is not required for Random Forest, we can still do it to reduce variables and complexity.

From the paiplots we can observe that: - Tuition and SPR.Group.Revenue has a correlation of 1 !! This means that both columns covey the same information so we can drop one. (On avg SPR.Group.Revenue has more corr with other so will drop it) - FPP and Total.Discount.Pax are also higly correlated with corr 0.837 - FPP and FRP_Actuals also have a high correlation of 0.811 - There are few other with 0.6 - 0.7 corr we won't be treating them now, we'll treat them after builing the base model and looking at important features.

- Also from the box plots above for RPL_diff we can see that the medians for both 1(Success) and 0(Fail) are at the same point, there also is no difference in it's distribution, since there are lot of missings in this columns and it isn't impacting the target, proceeding to drop it.

```
# Dropping columns with high correlation with other (FPP & SPR.Revenue)

drop_mcollinear <- c("FPP","SPR.Group.Revenue","RPL_diff")
dfn <- select(dfn, -drop_mcollinear)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(drop_mcollinear)‘ instead of ‘drop_mcollinear‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
introduce(dfn)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 2239      33               20                 13                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                  768          1875              73887       355144
```

```
num <- names(select(dfn, -cat))
num # new list of numerical column after dropping
```

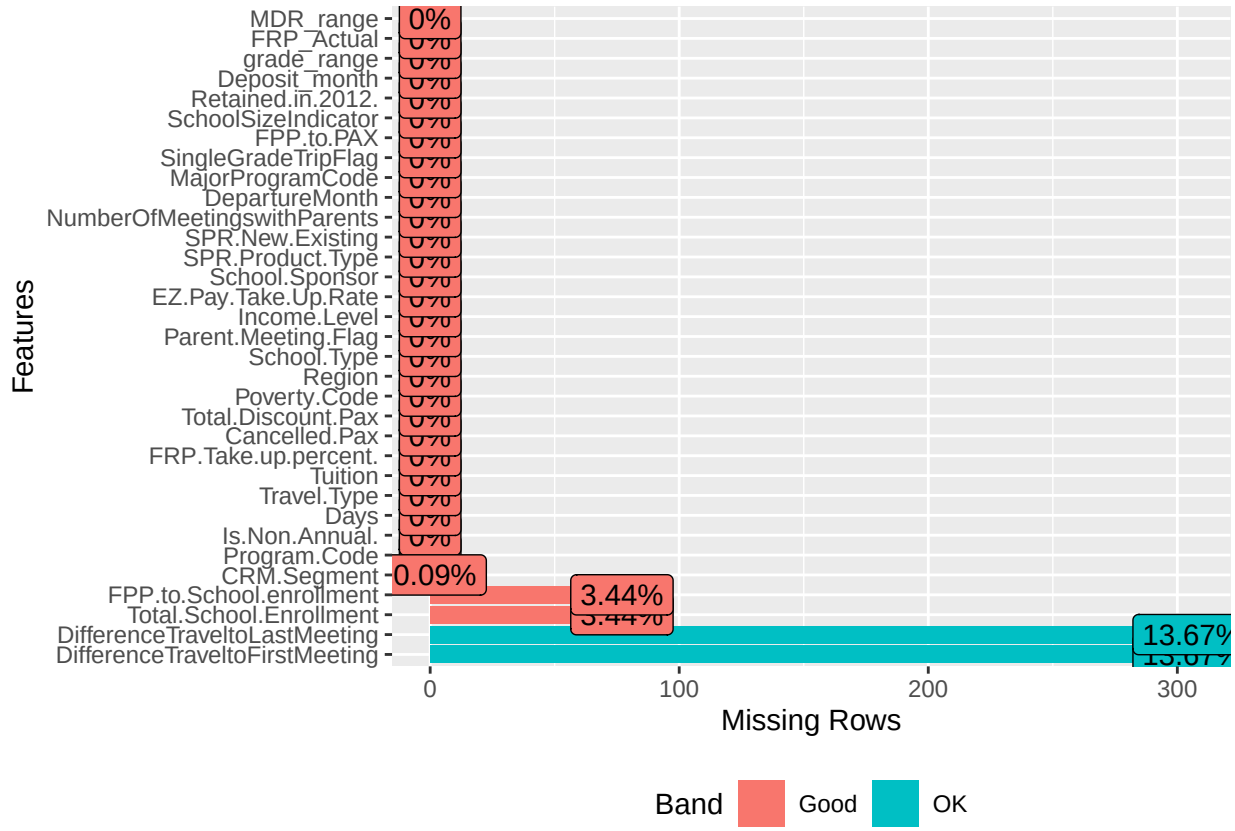
```
##  [1] "Days"                           "Tuition"
##  [3] "FRP.Take.up.percent."           "Cancelled.Pax"
##  [5] "Total.Discount.Pax"             "CRM.Segment"
##  [7] "Total.School.Enrollment"        "EZ.Pay.Take.Up.Rate"
##  [9] "DifferenceTraveltoFirstMeeting" "DifferenceTraveltoLastMeeting"
## [11] "FPP.to.School.enrollment"       "FPP.to.PAX"
## [13] "FRP_Actual"
```

MISSING VALUE IMPUTATION

```
# Checking missing values in each columns
introduce(dfn)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 2239      33               20                 13                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                  768          1875              73887       355144
```

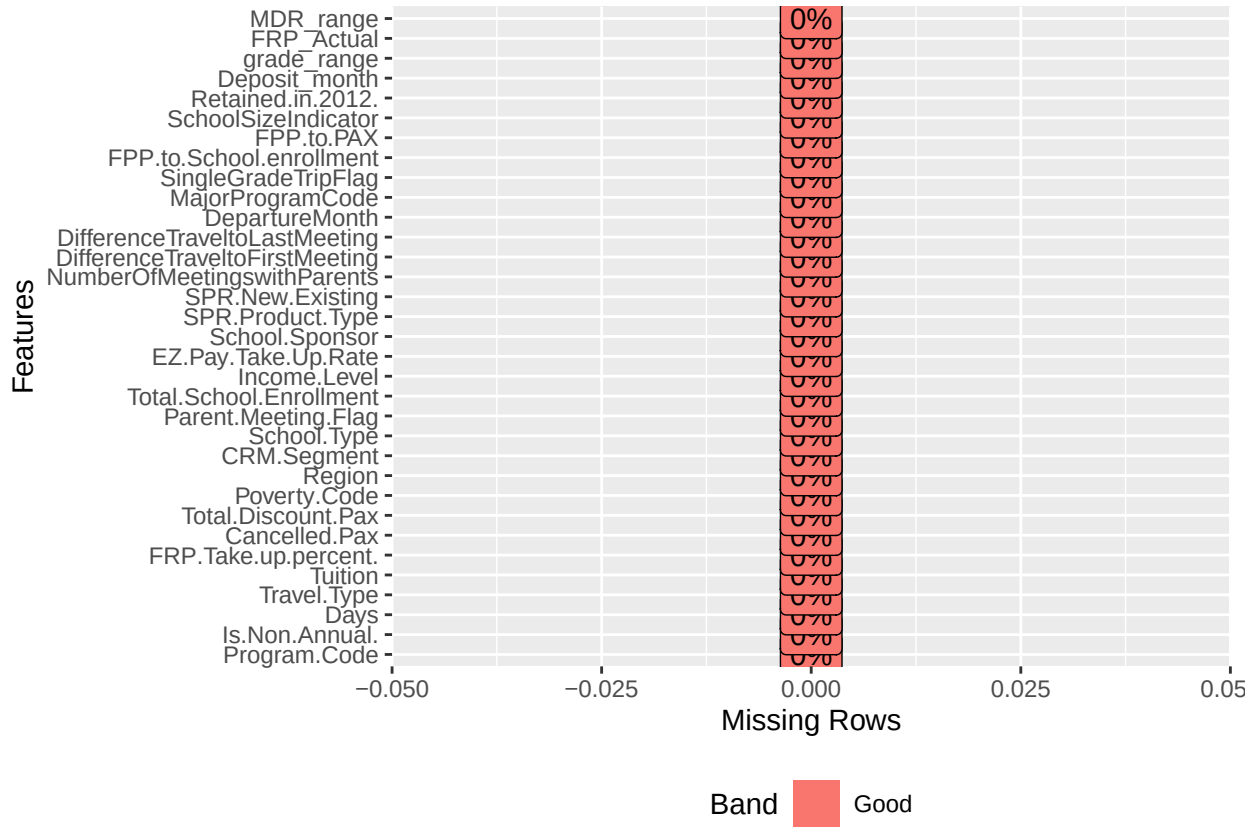
```
plot_missing(dfn)
```




```
# Plotting missing values and overview of data after imputation
introduce(dfi)
```

```
##   rows columns discrete_columns continuous_columns all_missing_columns
## 1 2239      33               20                 13                   0
##   total_missing_values complete_rows total_observations memory_usage
## 1                    0          2239               73887       355144
```

```
plot_missing(dfi)
```



Data has been cleaned so we can proceed to Model building.

MODEL BUILDING & TRAIN-TEST SPLIT

```
# Renaming target column for convenience
dfi <- dfi %>%
  rename( Target = Retained.in.2012.)
```

We'll do 80:20 split beacuse the dataset is small and we don't to end up with a underfit model due to less train data.

```
# Train-Test Split (80:20)
set.seed(5)
ind <- sample(2, nrow(dfi), replace= TRUE, prob = c(0.8, 0.2))
train <- dfi[ind == 1, ]
test <- dfi[ind == 2, ]

d_train <- list(dim(train))
d_test <- list(dim(test))

print(paste("Dimension of train data:",d_train))
```

```
## [1] "Dimension of train data: c(1776, 33)"
```

```r
print(paste("Dimension of test data:",d_test))
```

```
## [1] "Dimension of test data: c(463, 33)"
```

```r
# function to evaluate model with recall, precision and f-score :
metrics <- function(cm){
print(paste("Test accuracy :", sum(diag(cm)) / sum(cm)))
rc <- cm[2,2]/(cm[2,2]+cm[2,1])
pr <- cm[2,2]/(cm[2,2]+cm[1,2])
f <- 2*(pr*rc/(pr+rc))
print(paste("Recall of 1 (Success) :", rc))
print(paste("Precision of 1 (Success) :", pr))
print(paste("f score of 1 (Success) :", f))
}
```

```r
# function to display all performance metrics
full_metrics <- function(cm){
print(paste("Test accuracy :", sum(diag(cm)) / sum(cm)))

rc <- cm[2,2]/(cm[2,2]+cm[2,1])
pr <- cm[2,2]/(cm[2,2]+cm[1,2])
f <- 2*(pr*rc/(pr+rc))
sp <- cm[1,1]/(cm[1,1]+cm[1,2])
fpr <- cm[1,2]/(cm[1,1]+cm[1,2])
fnr <- cm[2,1]/(cm[2,2]+cm[2,1])

print(paste("Recall (Success) :", rc))
print(paste("Precision (Success) :", pr))
print(paste("F-score (Success) :", f))
print(paste("Specificity (tnr) :",sp))
print(paste("False positve rate:",fpr))
print(paste("False negative rate:",fnr))
}
```

DECISION TREE

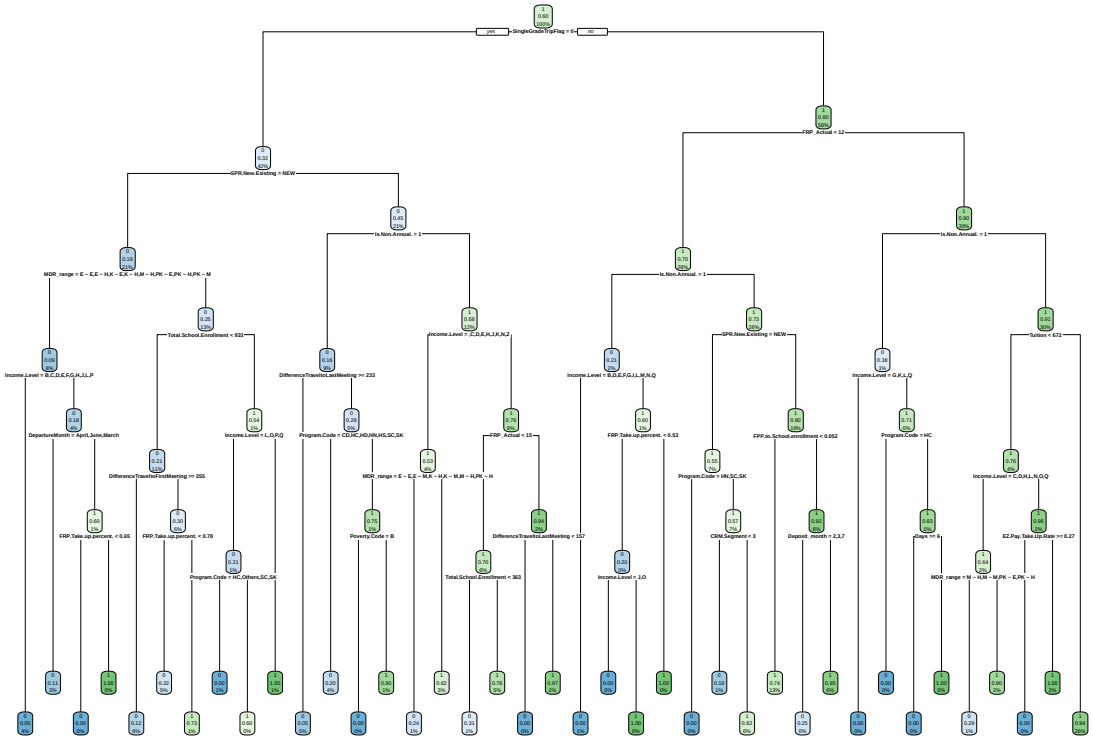
```r
tree_model <- rpart(Target ~ ., train, parms = list(split = "information"),
                    control = rpart.control(minbucket = 0, minsplit = 0,
                                            maxdepth = 6, cp =0))
print(tree_model)
```

```
## n= 1776
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 1776 707 1 (0.39808559 0.60191441)
##     2) SingleGradeTripFlag=0 743 238 0 (0.67967699 0.32032301)
##       4) SPR.New.Existing=NEW 368  69 0 (0.81250000 0.18750000)
```

```
##         8) MDR_range=E - E,E - H,K - E,K - H,M - H,PK - E,PK - H,PK - M 141  13 0 (0.90780142 0.09219
##          16) Income.Level=B,C,D,E,F,G,H,J,L,P 69   0 0 (1.00000000 0.00000000) *
##          17) Income.Level=A,I,K,M,N,O,Q,Z 72  13 0 (0.81944444 0.18055556)
##            34) DepartureMonth=April,June,March 62   7 0 (0.88709677 0.11290323) *
##            35) DepartureMonth=February,May 10   4 1 (0.40000000 0.60000000)
##              70) FRP.Take.up.percent.< 0.6465 4   0 0 (1.00000000 0.00000000) *
##              71) FRP.Take.up.percent.>=0.6465 6   0 1 (0.00000000 1.00000000) *
##         9) MDR_range= - NA,E - M,H - H,K - M,M - M 227  56 0 (0.75330396 0.24669604)
##          18) Total.School.Enrollment< 932.5 203  43 0 (0.78817734 0.21182266)
##            36) DifferenceTraveltoFirstMeeting>=254.5 98  12 0 (0.87755102 0.12244898) *
##            37) DifferenceTraveltoFirstMeeting< 254.5 105  31 0 (0.70476190 0.29523810)
##              74) FRP.Take.up.percent.< 0.7755 90  20 0 (0.77777778 0.22222222) *
##              75) FRP.Take.up.percent.>=0.7755 15   4 1 (0.26666667 0.73333333) *
##          19) Total.School.Enrollment>=932.5 24  11 1 (0.45833333 0.54166667)
##            38) Income.Level=L,O,P,Q 14   3 0 (0.78571429 0.21428571)
##              76) Program.Code=HC,Others,SC,SK 9   0 0 (1.00000000 0.00000000) *
##              77) Program.Code=HD 5   2 1 (0.40000000 0.60000000) *
##            39) Income.Level=B,C,F,I,J,K,M 10   0 1 (0.00000000 1.00000000) *
##       5) SPR.New.Existing=EXISTING 375 169 0 (0.54933333 0.45066667)
##        10) Is.Non.Annual.=1 167  27 0 (0.83832335 0.16167665)
##          20) DifferenceTraveltoLastMeeting>=233 86   4 0 (0.95348837 0.04651163) *
##          21) DifferenceTraveltoLastMeeting< 233 81  23 0 (0.71604938 0.28395062)
##            42) Program.Code=CD,HC,HD,HN,HS,SC,SK 69  14 0 (0.79710145 0.20289855) *
##            43) Program.Code=Others 12   3 1 (0.25000000 0.75000000)
##              86) Poverty.Code=B 2   0 0 (1.00000000 0.00000000) *
##              87) Poverty.Code=,A,C 10   1 1 (0.10000000 0.90000000) *
##        11) Is.Non.Annual.=0 208  66 1 (0.31730769 0.68269231)
##          22) Income.Level=,C,D,E,H,J,K,N,Z 70  33 1 (0.47142857 0.52857143)
##            44) MDR_range=E - E,E - M,K - H,K - M,M - H,PK - H 17   4 0 (0.76470588 0.23529412) *
##            45) MDR_range= - NA,K - E,M - M,PK - M 53  20 1 (0.37735849 0.62264151) *
##          23) Income.Level=B,F,G,I,L,M,O,P,Q 138  33 1 (0.23913043 0.76086957)
##            46) FRP_Actual< 14.5 104  31 1 (0.29807692 0.70192308)
##              92) Total.School.Enrollment< 363 13   4 0 (0.69230769 0.30769231) *
##              93) Total.School.Enrollment>=363 91  22 1 (0.24175824 0.75824176) *
##            47) FRP_Actual>=14.5 34   2 1 (0.05882353 0.94117647)
##              94) DifferenceTraveltoLastMeeting< 156.5 1   0 0 (1.00000000 0.00000000) *
##              95) DifferenceTraveltoLastMeeting>=156.5 33   1 1 (0.03030303 0.96969697) *
##     3) SingleGradeTripFlag=1 1033 202 1 (0.19554695 0.80445305)
##       6) FRP_Actual< 11.5 495 148 1 (0.29898990 0.70101010)
##        12) Is.Non.Annual.=1 28   6 0 (0.78571429 0.21428571)
##          24) Income.Level=B,D,E,F,G,I,L,M,N,Q 18   0 0 (1.00000000 0.00000000) *
##          25) Income.Level=H,J,K,O,P 10   4 1 (0.40000000 0.60000000)
##            50) FRP.Take.up.percent.< 0.533 5   1 0 (0.80000000 0.20000000)
##             100) Income.Level=J,O 4   0 0 (1.00000000 0.00000000) *
##             101) Income.Level=P 1   0 1 (0.00000000 1.00000000) *
##            51) FRP.Take.up.percent.>=0.533 5   0 1 (0.00000000 1.00000000) *
##        13) Is.Non.Annual.=0 467 126 1 (0.26980728 0.73019272)
##          26) SPR.New.Existing=NEW 126  57 1 (0.45238095 0.54761905)
##            52) Program.Code=HN,SC,SK 6   0 0 (1.00000000 0.00000000) *
##            53) Program.Code=CD,HC,HD,HS,HVP,Others 120  51 1 (0.42500000 0.57500000)
##             106) CRM.Segment< 2.5 10   1 0 (0.90000000 0.10000000) *
##             107) CRM.Segment>=2.5 110  42 1 (0.38181818 0.61818182) *
##          27) SPR.New.Existing=EXISTING 341  69 1 (0.20234604 0.79765396)
##            54) FPP.to.School.enrollment< 0.0521032 237  61 1 (0.25738397 0.74261603) *
```

```
##            55) FPP.to.School.enrollment>=0.0521032 104   8 1 (0.07692308 0.92307692)
##             110) Deposit_month=2,3,7 4   1 0 (0.75000000 0.25000000) *
##             111) Deposit_month=1,6,8,9,10,11,12 100   5 1 (0.05000000 0.95000000) *
##       7) FRP_Actual>=11.5 538  54 1 (0.10037175 0.89962825)
##        14) Is.Non.Annual.=1 13   5 0 (0.61538462 0.38461538)
##          28) Income.Level=G,K,L,Q 6   0 0 (1.00000000 0.00000000) *
##          29) Income.Level=,F,I,O,P 7   2 1 (0.28571429 0.71428571)
##            58) Program.Code=HC 1   0 0 (1.00000000 0.00000000) *
##            59) Program.Code=HD,Others 6   1 1 (0.16666667 0.83333333)
##             118) Days>=5.5 1   0 0 (1.00000000 0.00000000) *
##             119) Days< 5.5 5   0 1 (0.00000000 1.00000000) *
##        15) Is.Non.Annual.=0 525  46 1 (0.08761905 0.91238095)
##          30) Tuition< 672 72  17 1 (0.23611111 0.76388889)
##            60) Income.Level=C,D,H,L,N,O,Q 44  16 1 (0.36363636 0.63636364)
##             120) MDR_range=M - H,M - M,PK - E,PK - H 14   4 0 (0.71428571 0.28571429) *
##             121) MDR_range=E - M,K - E,K - H,K - M,PK - M 30   6 1 (0.20000000 0.80000000) *
##            61) Income.Level=,F,G,I,J,K,M,P,P1 28   1 1 (0.03571429 0.96428571)
##             122) EZ.Pay.Take.Up.Rate>=0.2655 1   0 0 (1.00000000 0.00000000) *
##             123) EZ.Pay.Take.Up.Rate< 0.2655 27   0 1 (0.00000000 1.00000000) *
##          31) Tuition>=672 453  29 1 (0.06401766 0.93598234) *
```

```
rpart.plot(tree_model)
```



```
pred_test <- predict(tree_model, test, type = "class")
cm_test <- table(test$Target, pred_test)
cm_test
```

```
##    pred_test
##       0   1
##   0  97  70
##   1  33 263
```

Pre-Pruning

For C5.0 tree:

```
bucket <- c(10,20,30)
split <- c(50,75,100)
for (i in bucket){
  for (j in split){
    print(paste("For bucket =",i,"and split =",j))
    tree_model2 <- rpart(Target ~ ., train, parms = list(split = "information"),
    control = rpart.control(minbucket = i, minsplit = j, maxdepth = 10, cp =0))
    pred_test <- predict(tree_model2, test, type = "class")
    cm_test <- table(test$Target, pred_test)
    metrics(cm_test)
    writeLines("\n\n")
  }
}
```

```
## [1] "For bucket = 10 and split = 50"
## [1] "Test accuracy : 0.781857451403888"
## [1] "Recall of 1 (Success) : 0.878378378378378"
## [1] "Precision of 1 (Success) : 0.8"
## [1] "f score of 1 (Success) : 0.837359098228664"
##
##
##
## [1] "For bucket = 10 and split = 75"
## [1] "Test accuracy : 0.788336933045356"
## [1] "Recall of 1 (Success) : 0.888513513513513"
## [1] "Precision of 1 (Success) : 0.801829268292683"
## [1] "f score of 1 (Success) : 0.842948717948718"
##
##
##
## [1] "For bucket = 10 and split = 100"
## [1] "Test accuracy : 0.784017278617711"
## [1] "Recall of 1 (Success) : 0.891891891891892"
## [1] "Precision of 1 (Success) : 0.795180722891566"
## [1] "f score of 1 (Success) : 0.840764331210191"
##
##
##
## [1] "For bucket = 20 and split = 50"
## [1] "Test accuracy : 0.781857451403888"
## [1] "Recall of 1 (Success) : 0.881756756756757"
## [1] "Precision of 1 (Success) : 0.798165137614679"
```

```
## [1] "f score of 1 (Success) : 0.837881219903692"
##
##
##
## [1] "For bucket = 20 and split = 75"
## [1] "Test accuracy : 0.781857451403888"
## [1] "Recall of 1 (Success) : 0.925675675675676"
## [1] "Precision of 1 (Success) : 0.776203966005666"
## [1] "f score of 1 (Success) : 0.844375963020031"
##
##
##
## [1] "For bucket = 20 and split = 100"
## [1] "Test accuracy : 0.781857451403888"
## [1] "Recall of 1 (Success) : 0.868243243243243"
## [1] "Precision of 1 (Success) : 0.80564263322884"
## [1] "f score of 1 (Success) : 0.835772357723577"
##
##
##
## [1] "For bucket = 30 and split = 50"
## [1] "Test accuracy : 0.775377969762419"
## [1] "Recall of 1 (Success) : 0.841216216216216"
## [1] "Precision of 1 (Success) : 0.813725490196078"
## [1] "f score of 1 (Success) : 0.827242524916944"
##
##
##
## [1] "For bucket = 30 and split = 75"
## [1] "Test accuracy : 0.779697624190065"
## [1] "Recall of 1 (Success) : 0.881756756756757"
## [1] "Precision of 1 (Success) : 0.795731707317073"
## [1] "f score of 1 (Success) : 0.836538461538462"
##
##
##
## [1] "For bucket = 30 and split = 100"
## [1] "Test accuracy : 0.786177105831533"
## [1] "Recall of 1 (Success) : 0.871621621621622"
## [1] "Precision of 1 (Success) : 0.808777429467085"
## [1] "f score of 1 (Success) : 0.839024390243902"
```

For C&R Tree:

```r
bucket <- c(10,20,30)
split <- c(50,75,100)
for (i in bucket){
  for (j in split){
    print(paste("For bucket =",i,"and split =",j))
    tree_model2 <- rpart(Target ~ ., train, parms = list(split = "gini"),
    control = rpart.control(minbucket = i, minsplit = j, maxdepth = 10, cp =0))
    pred_test <- predict(tree_model2, test, type = "class")
    cm_test <- table(test$Target, pred_test)
    metrics(cm_test)
```

```
    writeLines("\n\n")
  }
}
```

```
## [1] "For bucket = 10 and split = 50"
## [1] "Test accuracy : 0.773218142548596"
## [1] "Recall of 1 (Success) : 0.875"
## [1] "Precision of 1 (Success) : 0.792048929663609"
## [1] "f score of 1 (Success) : 0.831460674157303"
##
##
##
## [1] "For bucket = 10 and split = 75"
## [1] "Test accuracy : 0.786177105831533"
## [1] "Recall of 1 (Success) : 0.905405405405405"
## [1] "Precision of 1 (Success) : 0.790560471976401"
## [1] "f score of 1 (Success) : 0.844094488188976"
##
##
##
## [1] "For bucket = 10 and split = 100"
## [1] "Test accuracy : 0.788336933045356"
## [1] "Recall of 1 (Success) : 0.864864864864865"
## [1] "Precision of 1 (Success) : 0.815286624203822"
## [1] "f score of 1 (Success) : 0.839344262295082"
##
##
##
## [1] "For bucket = 20 and split = 50"
## [1] "Test accuracy : 0.788336933045356"
## [1] "Recall of 1 (Success) : 0.878378378378378"
## [1] "Precision of 1 (Success) : 0.807453416149068"
## [1] "f score of 1 (Success) : 0.841423948220065"
##
##
##
## [1] "For bucket = 20 and split = 75"
## [1] "Test accuracy : 0.799136069114471"
## [1] "Recall of 1 (Success) : 0.898648648648649"
## [1] "Precision of 1 (Success) : 0.808510638297872"
## [1] "f score of 1 (Success) : 0.8512"
##
##
##
## [1] "For bucket = 20 and split = 100"
## [1] "Test accuracy : 0.807775377969762"
## [1] "Recall of 1 (Success) : 0.881756756756757"
## [1] "Precision of 1 (Success) : 0.828571428571429"
## [1] "f score of 1 (Success) : 0.854337152209493"
##
##
##
## [1] "For bucket = 30 and split = 50"
```

```
## [1] "Test accuracy : 0.807775377969762"
## [1] "Recall of 1 (Success) : 0.864864864864865"
## [1] "Precision of 1 (Success) : 0.839344262295082"
## [1] "f score of 1 (Success) : 0.851913477537438"
##
##
##
## [1] "For bucket = 30 and split = 75"
## [1] "Test accuracy : 0.809935205183585"
## [1] "Recall of 1 (Success) : 0.881756756756757"
## [1] "Precision of 1 (Success) : 0.831210191082803"
## [1] "f score of 1 (Success) : 0.855737704918033"
##
##
##
## [1] "For bucket = 30 and split = 100"
## [1] "Test accuracy : 0.812095032397408"
## [1] "Recall of 1 (Success) : 0.875"
## [1] "Precision of 1 (Success) : 0.838187702265372"
## [1] "f score of 1 (Success) : 0.856198347107438"
```

Best parameter: - C&R Tree (Gini) - for C&R, minbucket = 30 - for C&R, minsplit = 100

```r
# Further Tuning Tree:
bucket <- c(30,35,40)
split <- c(100,150,200,250)
for (i in bucket){
  for (j in split){
    print(paste("For bucket =",i,"and split =",j))
    tree_model2 <- rpart(Target ~ ., train, parms = list(split = "gini"),
    control = rpart.control(minbucket = i, minsplit = j, maxdepth = 10, cp =0))
    pred_test <- predict(tree_model2, test, type = "class")
    cm_test <- table(test$Target, pred_test)
    metrics(cm_test)
    writeLines("\n\n")
  }
}
```

```
## [1] "For bucket = 30 and split = 100"
## [1] "Test accuracy : 0.812095032397408"
## [1] "Recall of 1 (Success) : 0.875"
## [1] "Precision of 1 (Success) : 0.838187702265372"
## [1] "f score of 1 (Success) : 0.856198347107438"
##
##
##
## [1] "For bucket = 30 and split = 150"
## [1] "Test accuracy : 0.812095032397408"
## [1] "Recall of 1 (Success) : 0.875"
## [1] "Precision of 1 (Success) : 0.838187702265372"
## [1] "f score of 1 (Success) : 0.856198347107438"
##
##
##
```

```
## [1] "For bucket = 30 and split = 200"
## [1] "Test accuracy : 0.818574514038877"
## [1] "Recall of 1 (Success) : 0.922297297297297"
## [1] "Precision of 1 (Success) : 0.817365269461078"
## [1] "f score of 1 (Success) : 0.866666666666667"
##
##
##
## [1] "For bucket = 30 and split = 250"
## [1] "Test accuracy : 0.816414686825054"
## [1] "Recall of 1 (Success) : 0.925675675675676"
## [1] "Precision of 1 (Success) : 0.813056379821958"
## [1] "f score of 1 (Success) : 0.865718799368088"
##
##
##
## [1] "For bucket = 35 and split = 100"
## [1] "Test accuracy : 0.809935205183585"
## [1] "Recall of 1 (Success) : 0.878378378378378"
## [1] "Precision of 1 (Success) : 0.833333333333333"
## [1] "f score of 1 (Success) : 0.855263157894737"
##
##
##
## [1] "For bucket = 35 and split = 150"
## [1] "Test accuracy : 0.809935205183585"
## [1] "Recall of 1 (Success) : 0.878378378378378"
## [1] "Precision of 1 (Success) : 0.833333333333333"
## [1] "f score of 1 (Success) : 0.855263157894737"
##
##
##
## [1] "For bucket = 35 and split = 200"
## [1] "Test accuracy : 0.816414686825054"
## [1] "Recall of 1 (Success) : 0.925675675675676"
## [1] "Precision of 1 (Success) : 0.813056379821958"
## [1] "f score of 1 (Success) : 0.865718799368088"
##
##
##
## [1] "For bucket = 35 and split = 250"
## [1] "Test accuracy : 0.816414686825054"
## [1] "Recall of 1 (Success) : 0.925675675675676"
## [1] "Precision of 1 (Success) : 0.813056379821958"
## [1] "f score of 1 (Success) : 0.865718799368088"
##
##
##
## [1] "For bucket = 40 and split = 100"
## [1] "Test accuracy : 0.809935205183585"
## [1] "Recall of 1 (Success) : 0.878378378378378"
## [1] "Precision of 1 (Success) : 0.833333333333333"
## [1] "f score of 1 (Success) : 0.855263157894737"
##
```

```
##
##
## [1] "For bucket = 40 and split = 150"
## [1] "Test accuracy : 0.809935205183585"
## [1] "Recall of 1 (Success) : 0.878378378378378"
## [1] "Precision of 1 (Success) : 0.833333333333333"
## [1] "f score of 1 (Success) : 0.855263157894737"
##
##
##
## [1] "For bucket = 40 and split = 200"
## [1] "Test accuracy : 0.816414686825054"
## [1] "Recall of 1 (Success) : 0.925675675675676"
## [1] "Precision of 1 (Success) : 0.813056379821958"
## [1] "f score of 1 (Success) : 0.865718799368088"
##
##
##
## [1] "For bucket = 40 and split = 250"
## [1] "Test accuracy : 0.816414686825054"
## [1] "Recall of 1 (Success) : 0.925675675675676"
## [1] "Precision of 1 (Success) : 0.813056379821958"
## [1] "f score of 1 (Success) : 0.865718799368088"
```

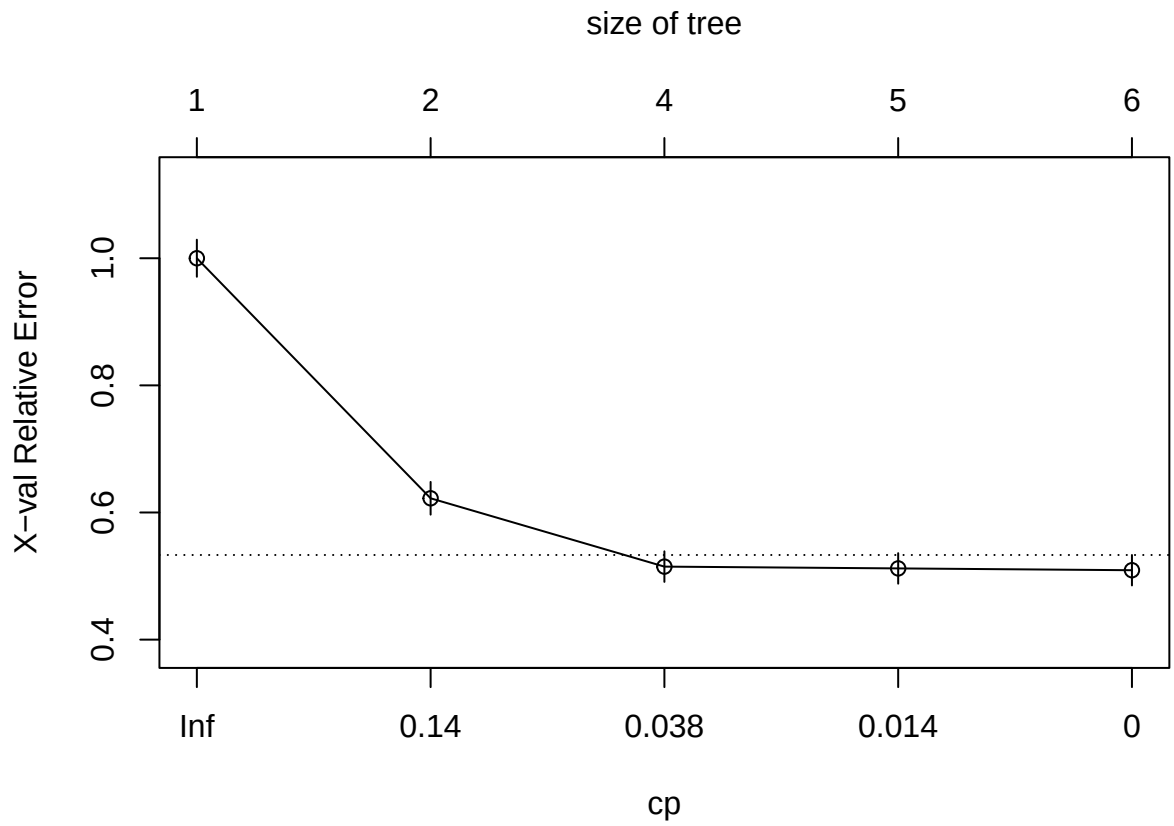
Best parameters for the tree are : (30,200) for minbucket and minsample

Post - Pruning

```r
# Determining best cp value for best minbucket and minsplit
tree_model_tune <- rpart(Target ~ ., train, parms = list(split = "gini"),
            control = rpart.control(minbucket = 30, minsplit = 200, cp=0))
printcp(tree_model_tune)
```

```
##
## Classification tree:
## rpart(formula = Target ~ ., data = train, parms = list(split = "gini"),
##     control = rpart.control(minbucket = 30, minsplit = 200, cp = 0))
##
## Variables actually used in tree construction:
## [1] Is.Non.Annual.          SingleGradeTripFlag     SPR.New.Existing
## [4] Total.School.Enrollment
##
## Root node error: 707/1776 = 0.39809
##
## n= 1776
##
##          CP nsplit rel error  xerror     xstd
## 1 0.3776521      0   1.00000 1.00000 0.029178
## 2 0.0537482      1   0.62235 0.62235 0.025733
## 3 0.0268741      3   0.51485 0.51485 0.024062
## 4 0.0070721      4   0.48798 0.51202 0.024013
## 5 0.0000000      5   0.48091 0.50919 0.023963
```

```r
plotcp(tree_model_tune)
```

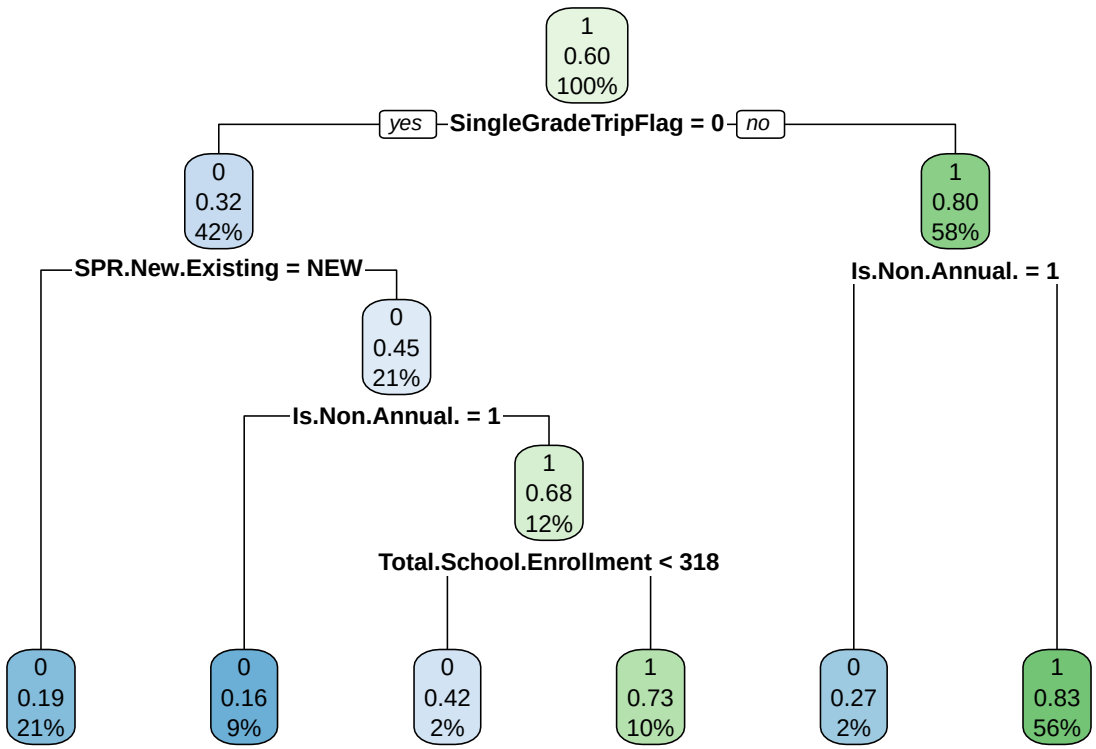


Best cp value from table = 0.0070721

```r
cp <- 0.0070721
prunedTree <- tree_model_tune <- rpart(Target ~ ., train, parms = list(split = "gini"),
            control = rpart.control(minbucket = 30, minsplit = 200, cp=cp))
print(prunedTree)
```

```
## n= 1776
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 1776 707 1 (0.3980856 0.6019144)
##    2) SingleGradeTripFlag=0 743 238 0 (0.6796770 0.3203230)
##      4) SPR.New.Existing=NEW 368  69 0 (0.8125000 0.1875000) *
##      5) SPR.New.Existing=EXISTING 375 169 0 (0.5493333 0.4506667)
##       10) Is.Non.Annual.=1 167  27 0 (0.8383234 0.1616766) *
##       11) Is.Non.Annual.=0 208  66 1 (0.3173077 0.6826923)
##         22) Total.School.Enrollment< 318 31  13 0 (0.5806452 0.4193548) *
##         23) Total.School.Enrollment>=318 177  48 1 (0.2711864 0.7288136) *
##    3) SingleGradeTripFlag=1 1033 202 1 (0.1955470 0.8044530)
##      6) Is.Non.Annual.=1 41  11 0 (0.7317073 0.2682927) *
##      7) Is.Non.Annual.=0 992 172 1 (0.1733871 0.8266129) *
```

```
rpart.plot(prunedTree)
```




```
# Checking classification nmetrics after post pruning
pred_test_prune <- predict(prunedTree, test, type = "class")
cm_test_prune <- table(test$Target, pred_test_prune)
metrics(cm_test_prune)
```

```
## [1] "Test accuracy : 0.818574514038877"
## [1] "Recall of 1 (Success) : 0.922297297297297"
## [1] "Precision of 1 (Success) : 0.817365269461078"
## [1] "f score of 1 (Success) : 0.866666666666667"
```

We can see that there isn't much improvement after post pruning, this might be because the tree was already tuned during pre-prunning.

RANDOM FOREST

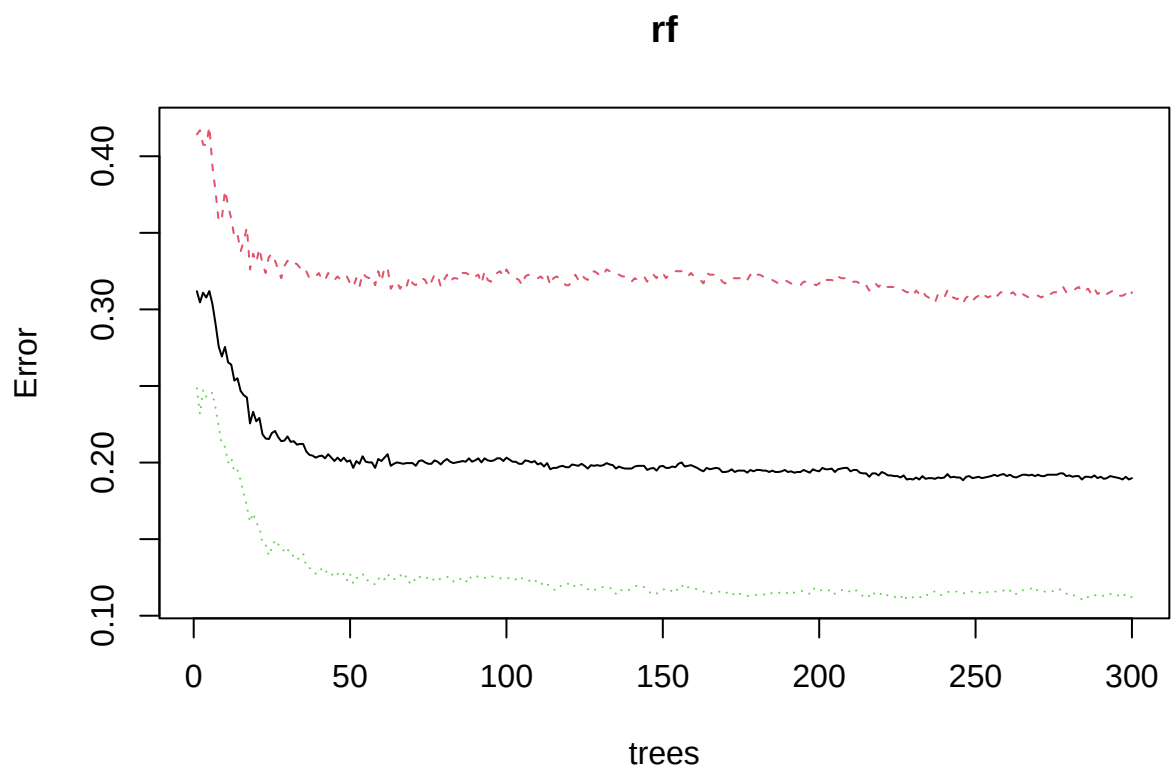
Base Model :

```
set.seed(5)
rf <- randomForest(Target ~ ., data = dfi, mtry = sqrt(ncol(dfi)-1),
                   ntree = 300, proximity = T, importance = T)
print(rf)
```

```
##
## Call:
##  randomForest(formula = Target ~ ., data = dfi, mtry = sqrt(ncol(dfi) -      1), ntree = 300, proxim
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 18.98%
## Confusion matrix:
##     0    1 class.error
## 0 602  272   0.3112128
## 1 153 1212   0.1120879
```

We can see OOB error of ~19% which is not bad

```
plot(rf)
```



We can see the error rate saturating at around n=150 to 300

Important Features

```
importance(rf, type = 1)
```

```
##                                   MeanDecreaseAccuracy
```

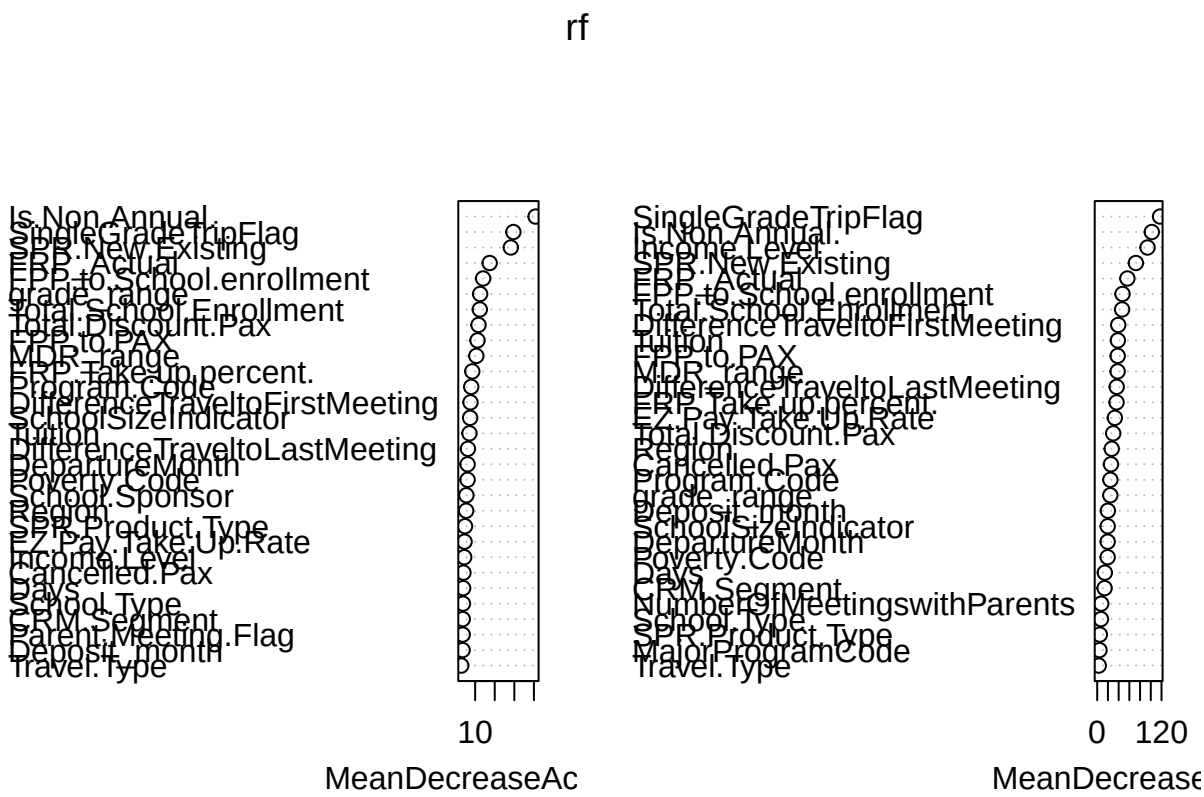
```
## Program.Code                                7.983136
## Is.Non.Annual.                             40.877558
## Days                                        3.862632
## Travel.Type                                 2.904201
## Tuition                                     7.108867
## FRP.Take.up.percent.                        8.527713
## Cancelled.Pax                               4.021262
## Total.Discount.Pax                         11.718968
## Poverty.Code                                6.132204
## Region                                      5.410348
## CRM.Segment                                 3.676412
## School.Type                                 3.785007
## Parent.Meeting.Flag                         3.650940
## Total.School.Enrollment                    12.371088
## Income.Level                                4.401539
## EZ.Pay.Take.Up.Rate                         4.587918
## School.Sponsor                              5.519106
## SPR.Product.Type                            4.818985
## SPR.New.Existing                           28.206408
## NumberOfMeetingswithParents                 1.426576
## DifferenceTraveltoFirstMeeting              7.625845
## DifferenceTraveltoLastMeeting               6.289009
## DepartureMonth                              6.137325
## MajorProgramCode                            2.741500
## SingleGradeTripFlag                        29.499610
## FPP.to.School.enrollment                   14.067317
## FPP.to.PAX                                 11.207914
## SchoolSizeIndicator                         7.436951
## Deposit_month                               3.584254
## grade_range                                12.536012
## FRP_Actual                                 17.417870
## MDR_range                                  10.599432
```

```
importance(rf, type = 2)
```

```
##                                  MeanDecreaseGini
## Program.Code                            24.552090
## Is.Non.Annual.                         101.879094
## Days                                    14.233220
## Travel.Type                              2.896297
## Tuition                                 38.505167
## FRP.Take.up.percent.                    36.098510
## Cancelled.Pax                           25.612115
## Total.Discount.Pax                      29.860072
## Poverty.Code                            19.512175
## Region                                  27.054507
## CRM.Segment                             13.813362
## School.Type                              6.993356
## Parent.Meeting.Flag                      2.725191
## Total.School.Enrollment                 46.444336
## Income.Level                            93.895100
## EZ.Pay.Take.Up.Rate                     32.918341
## School.Sponsor                           2.727325
## SPR.Product.Type                         4.928528
```

```
## SPR.New.Existing                      72.166317
## NumberOfMeetingswithParents            7.468319
## DifferenceTraveltoFirstMeeting        39.253778
## DifferenceTraveltoLastMeeting         36.375878
## DepartureMonth                        19.540094
## MajorProgramCode                       4.210866
## SingleGradeTripFlag                  117.345655
## FPP.to.School.enrollment              47.233978
## FPP.to.PAX                            38.224184
## SchoolSizeIndicator                   19.569213
## Deposit_month                         19.906488
## grade_range                           24.460278
## FRP_Actual                            56.311840
## MDR_range                             38.068932
```

```
varImpPlot(rf)
```



From the above plot and table we can infer that the important variables are: - Is.Non.Annual - SingleGradeTripFlag - Spr.New.Existing - FPP.to.School.enrollement - Total.Discount.pax We can also see that our feature extracted variables also have a relatively high importance, hence it was worthwile to do the extractions.

Tuning Random Forest

Finding best value of mtry for getting highest fscore:

```r
set.seed(9)
f_sc <- c()
for(mt in seq(6,ncol(train)-6)){
  rf1 <- randomForest(Target~., data = train, ntree = 200, mtry =  mt)

  pred_test_rf <- predict(rf1, test, type = "class")
  cm <- table(test$Target, pred_test_rf)

  rc <- cm[2,2]/(cm[2,2]+cm[2,1])
  pr <- cm[2,2]/(cm[2,2]+cm[1,2])
  fs <- 2*(pr*rc/(pr+rc))

  f_sc <- c(f_sc,fs)
  print(paste("F score for mtry -",mt," is: ",fs))
}
```

```
## [1] "F score for mtry - 6  is:  0.866242038216561"
## [1] "F score for mtry - 7  is:  0.867088607594937"
## [1] "F score for mtry - 8  is:  0.868044515103339"
## [1] "F score for mtry - 9  is:  0.862619808306709"
## [1] "F score for mtry - 10  is:  0.864433811802233"
## [1] "F score for mtry - 11  is:  0.865814696485623"
## [1] "F score for mtry - 12  is:  0.863492063492063"
## [1] "F score for mtry - 13  is:  0.870813397129187"
## [1] "F score for mtry - 14  is:  0.863563402889246"
## [1] "F score for mtry - 15  is:  0.869426751592357"
## [1] "F score for mtry - 16  is:  0.86634460547504"
## [1] "F score for mtry - 17  is:  0.870813397129187"
## [1] "F score for mtry - 18  is:  0.869009584664537"
## [1] "F score for mtry - 19  is:  0.8736"
## [1] "F score for mtry - 20  is:  0.870813397129187"
## [1] "F score for mtry - 21  is:  0.87758346581876"
## [1] "F score for mtry - 22  is:  0.8704"
## [1] "F score for mtry - 23  is:  0.874003189792663"
## [1] "F score for mtry - 24  is:  0.869426751592357"
## [1] "F score for mtry - 25  is:  0.8672"
## [1] "F score for mtry - 26  is:  0.872611464968153"
## [1] "F score for mtry - 27  is:  0.868044515103339"
```

```r
bestmtry <- which.max(f_sc)+5

print(paste("Best value for mtry :",bestmtry))
```

```
## [1] "Best value for mtry : 21"
```

```r
set.seed(9)
rf_tune <- randomForest(Target~., data = train, ntree = 200, mtry =  bestmtry)
pred_test_rf <- predict(rf_tune, test, type = "class")
cm_rft <- table(test$Target, pred_test_rf)
print(rf1)
```

```
##
```

```
## Call:
##  randomForest(formula = Target ~ ., data = train, ntree = 200,      mtry = mt)
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 20.38%
## Confusion matrix:
##     0   1 class.error
## 0 479 228   0.3224894
## 1 134 935   0.1253508
```

```
writeLines("\n")
```

```
print("Test Confusion Matrix: ")
```

```
## [1] "Test Confusion Matrix: "
```

```
print(cm_rft)
```

```
##    pred_test_rf
##       0   1
##   0 110  57
##   1  23 273
```

```
writeLines("\n")
```

```
metrics(cm_rft)
```

```
## [1] "Test accuracy : 0.827213822894168"
## [1] "Recall of 1 (Success) : 0.922297297297297"
## [1] "Precision of 1 (Success) : 0.827272727272727"
## [1] "f score of 1 (Success) : 0.87220447284345"
```

We can see a slight improvement in Our Tuned Random Forest Model than our tuned Decision tree. The fscore has improved from ~8.66 to ~8.72

K FOLD Cross Validation

```
set.seed(4)
dfi <- dfi[sample(nrow(dfi)), ] #Shufffling rows

k <- 10
nmethod <- 2
folds <- cut(seq(1,nrow(dfi)), breaks=k, labels=FALSE)
model_err <- matrix(-1, k, nmethod, dimnames= list(paste0("Folds ", 1:k),
                                c("DecisionTree","RandomForest")))
```

```r
for (i in 1:k){
  test_ind <- which(folds==i, arr.ind = TRUE)
  test_cv <- dfi[test_ind, ]
  train_cv <- dfi[-test_ind, ]

  CVtree <- rpart(Target ~ ., train_cv, parms = list(split = "gini"),
            control = rpart.control(minbucket = 30, minsplit = 200, cp=cp))
  dt_pred <- predict(CVtree, test_cv, type='class')
  cm_dt <- table(test_cv$Target, dt_pred)
  rc_dt <- cm_dt[2,2]/(cm_dt[2,2]+cm_dt[2,1])
  pr_dt <- cm_dt[2,2]/(cm_dt[2,2]+cm_dt[1,2])
  fs_dt <- 2*(pr_dt*rc_dt/(pr_dt+rc_dt))

  CVrf <- randomForest(Target~., data = train_cv, ntree = 300, mtry =  bestmtry)
  rf_pred <- predict(CVrf, test_cv, type='class')
  cm_rf <- table(test_cv$Target, rf_pred)
  rc_rf <- cm_rf[2,2]/(cm_rf[2,2]+cm_rf[2,1])
  pr_rf <- cm_rf[2,2]/(cm_rf[2,2]+cm_rf[1,2])
  fs_rf <- 2*(pr_rf*rc_rf/(pr_rf+rc_rf))

  model_err[i,1] <- fs_dt
  model_err[i,2] <- fs_rf

}

model_err
```

```
##          DecisionTree RandomForest
## Folds 1     0.8636364    0.8659794
## Folds 2     0.8611111    0.8602941
## Folds 3     0.8494983    0.8533333
## Folds 4     0.8728522    0.8513514
## Folds 5     0.8474576    0.8474576
## Folds 6     0.8541667    0.8464164
## Folds 7     0.8400000    0.8247423
## Folds 8     0.8389513    0.8473282
## Folds 9     0.8072727    0.8118081
## Folds 10    0.8442907    0.8387097
```

```r
print(paste("Mean fscore for Decision Tree: ",mean(model_err[,1])))
```

```
## [1] "Mean fscore for Decision Tree:  0.847923702554256"
```

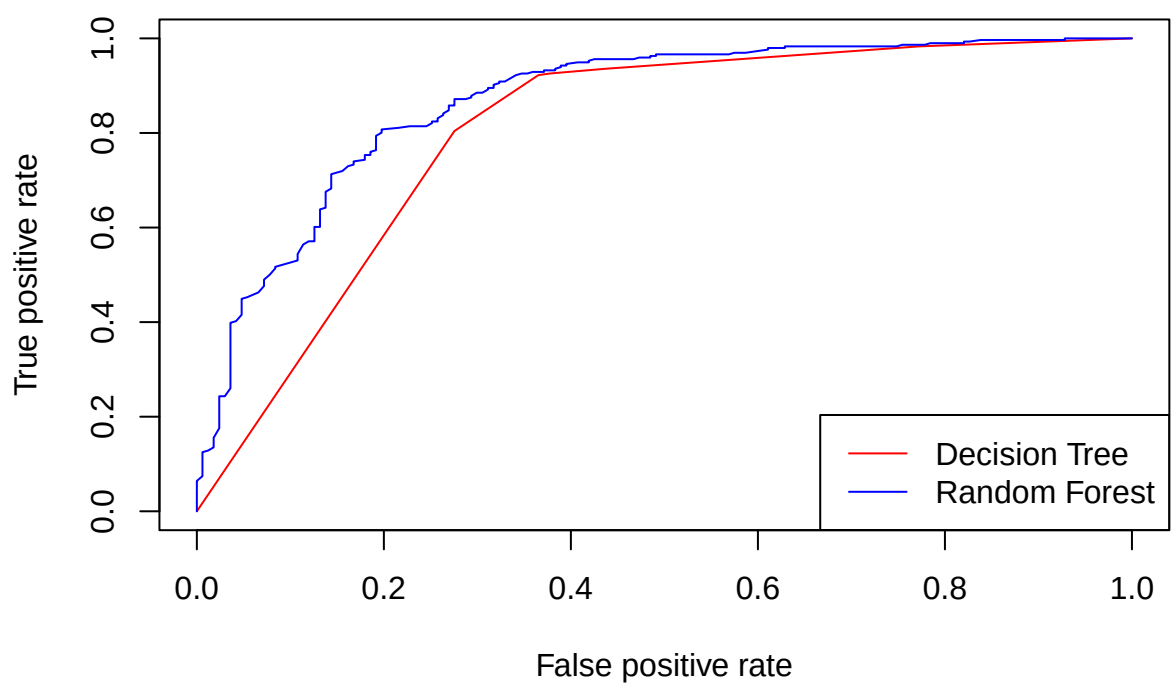
```r
print(paste("Mean fscore for Random Forest: ",mean(model_err[,2])))
```

```
## [1] "Mean fscore for Random Forest:  0.844742050096283"
```

Again we can see that there's very minor difference between the 2 tuned models Let's plot the ROC curve for both and check which performs better

COMPARING MODELS (ROC CURVES & CUT-OFF)

```
CVtree <- rpart(Target ~ ., train, parms = list(split = "gini"),
            control = rpart.control(minbucket = 30, minsplit = 200, cp=cp))
dt_score <- predict(CVtree, test, type='prob')
dt_pred <- prediction(dt_score[,2], test$Target)


CVrf <- randomForest(Target~., data = train, ntree = 300, mtry =  bestmtry)
rf_score <- predict(CVrf, test, type='prob')
rf_pred <- prediction(rf_score[,2], test$Target)

perf_dt <- performance(dt_pred, "tpr", "fpr")
perf_rf <- performance(rf_pred, "tpr", "fpr")


plot(perf_dt, col="red")
plot(perf_rf, add = TRUE, col="blue")
legend("bottomright", c("Decision Tree", "Random Forest"), lty=1,
    col = c("red", "blue"))
```



From the ROC Curves we can conclude that our tuned Random Forest Model is performing better than our tuned Decision Tree model. This also coincides with our earlier findings.

Hence, let's fix Random Forest as our final model and get the best cut-off points/threshold value.

```r
# Function to get cut-off points :
opt.cut <- function(perf){
  cut.ind <- mapply(FUN = function(x,y,p){
    d=(x-0)^2+(y-1)^2
    ind<- which(d==min(d))
    c(recall = y[[ind]], specificity = 1-x[[ind]],cutoff = p[[ind]])
    },
  perf@x.values, perf@y.values,perf@alpha.values)
}
```

```r
set.seed(9)
print("Cut-off points :")
```

```
## [1] "Cut-off points :"
```

```r
print(paste("tpr :",opt.cut(perf_rf)[1]))
```

```
## [1] "tpr : 0.807432432432432"
```

```r
print(paste("fpr :",1-opt.cut(perf_rf)[2]))
```

```
## [1] "fpr : 0.197604790419162"
```

```r
print(paste("Best Threshold :",opt.cut(perf_rf)[3]))
```

```
## [1] "Best Threshold : 0.673333333333333"
```

```r
bst_thr <- opt.cut(perf_rf)[3]
```

FINAL MODEL AND EVALUATION CHARTS

We'll build a Random Forest model using the above threshold and parameters which we tuned earlier to create the best/final model.

```r
rf_final <- randomForest(Target~., data = train, ntree = 300, mtry =  bestmtry)
rf_final_score <- predict(rf_final, test, type='prob')
rf_final_class <- predict(rf_final, test, type='class')
rf_final_class_thresh <- as.factor(ifelse(rf_final_score[,2]>=bst_thr,"1","0"))
rf_final_pred <- prediction(rf_final_score[,2], test$Target)

cm_rf_final_thresh  <- table(test$Target, rf_final_class_thresh)
cm_rf_final <- table(test$Target, rf_final_class)

print("Confusion Matix and other metrics before implemenying cut-off threshold")
```

```
## [1] "Confusion Matix and other metrics before implemenying cut-off threshold"
```

```r
full_metrics(cm_rf_final)
```

```
## [1] "Test accuracy : 0.827213822894168"
## [1] "Recall (Success) : 0.922297297297297"
## [1] "Precision (Success) : 0.827272727272727"
## [1] "F-score (Success) : 0.87220447284345"
## [1] "Specificity (tnr) : 0.658682634730539"
## [1] "False positve rate: 0.341317365269461"
## [1] "False negative rate: 0.0777027027027027"
```

```r
print("----------------------------------------------------------------------")
```

```
## [1] "----------------------------------------------------------------------"
```

```r
print("Confusion matrix and other performance metrics after implementing threshold")
```

```
## [1] "Confusion matrix and other performance metrics after implementing threshold"
```

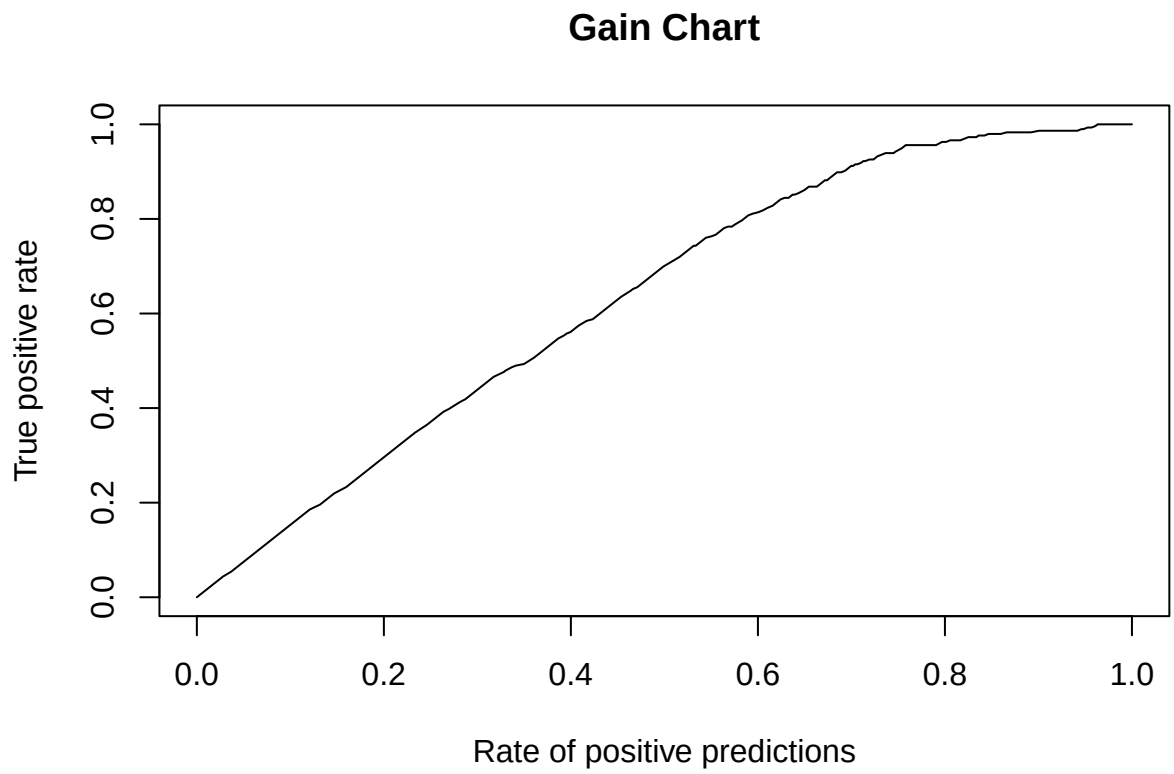
```r
full_metrics(cm_rf_final_thresh)
```

```
## [1] "Test accuracy : 0.799136069114471"
## [1] "Recall (Success) : 0.800675675675676"
## [1] "Precision (Success) : 0.874538745387454"
## [1] "F-score (Success) : 0.835978835978836"
## [1] "Specificity (tnr) : 0.796407185628742"
## [1] "False positve rate: 0.203592814371257"
## [1] "False negative rate: 0.199324324324324"
```

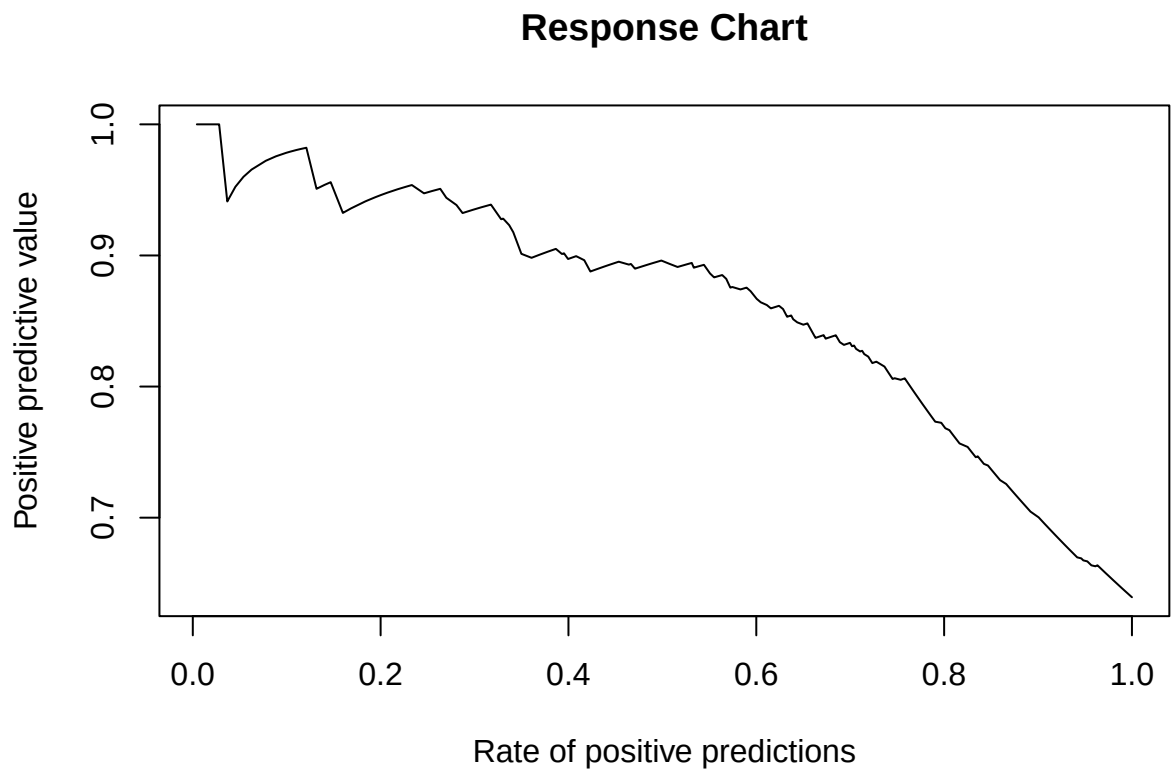
WE can see a slight drop in performance after implmenting the cut-off threshold obatined from ROC plot, this might possible because of the fine tuning of hyper-paramenters that we did earlier. Since they were tuned with respect to default threshold, there is a slight drop in performance after changing the threshold value. Also another thing to notice is our specificity has improved after applying the threshold, this nakes sence since we optimized it with respect to ROC curve which is recall vs 1-specificity

Evaluation Charts:

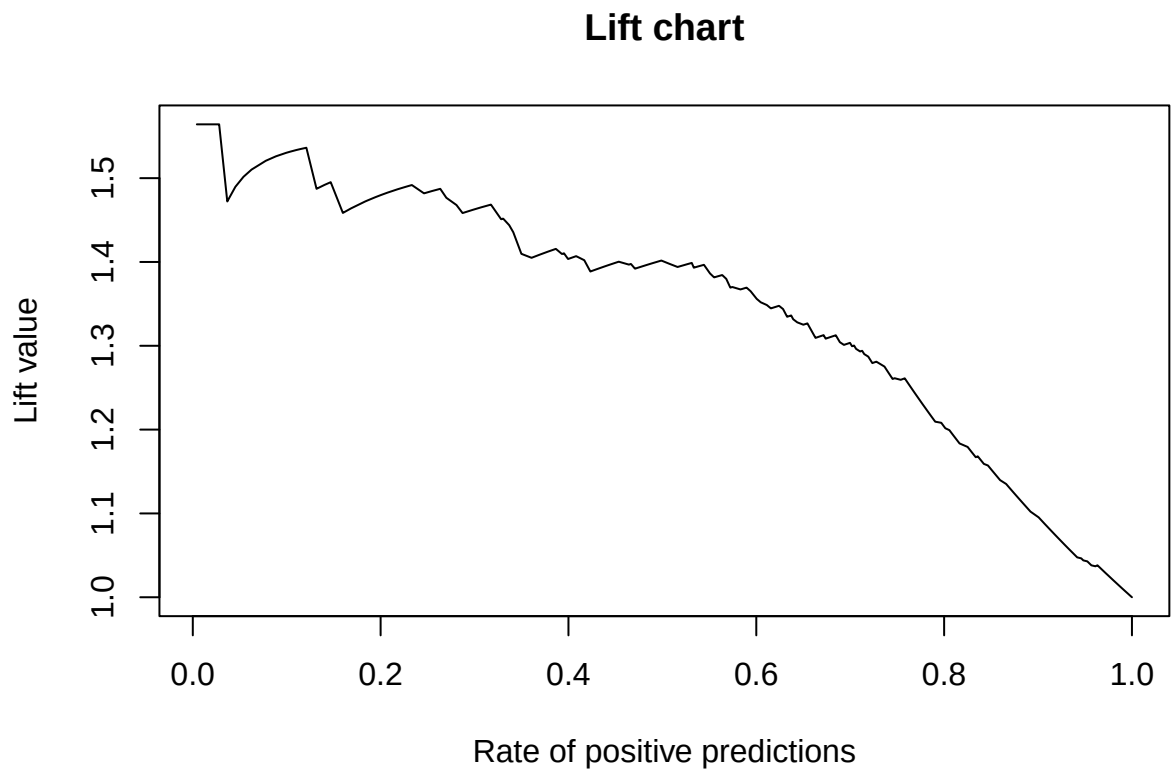
```r
# Gain Chart
perf <- performance(rf_final_pred, "tpr", "rpp")
plot(perf, main="Gain Chart")
```



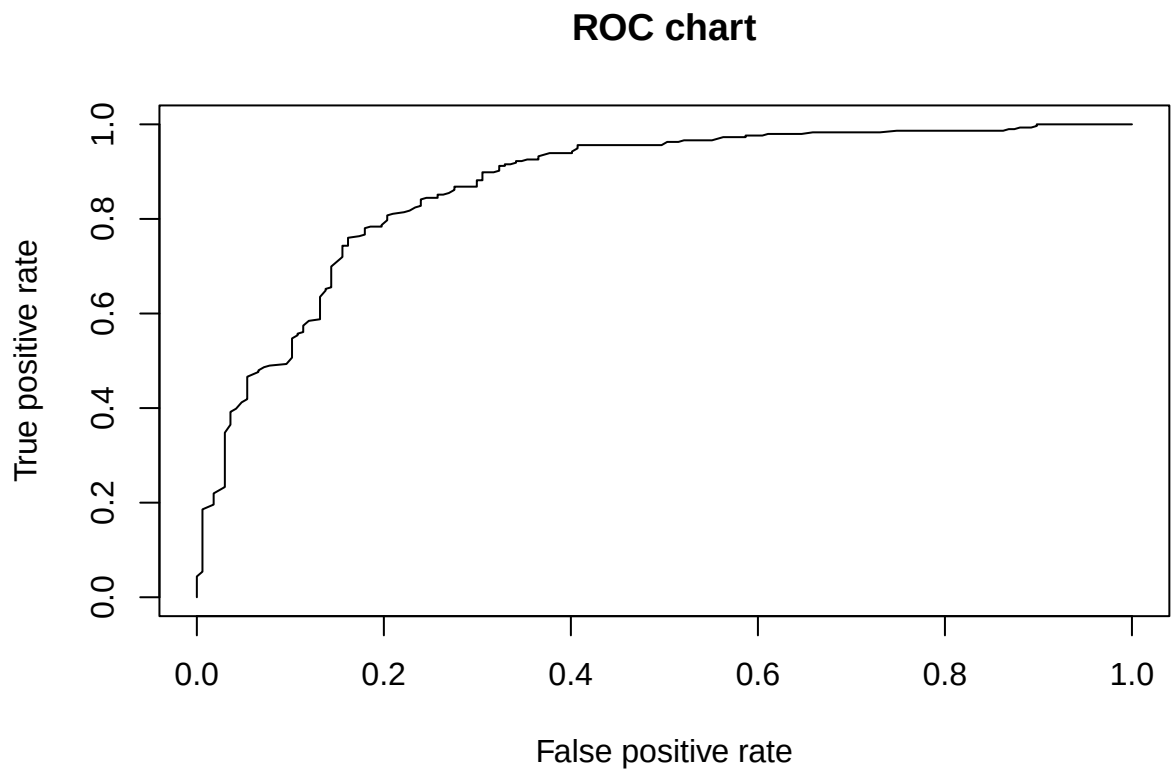

```
# Response Chart
perf <- performance(rf_final_pred, "ppv", "rpp")
plot(perf, main='Response Chart')
```




```
# Lift Chart
perf <- performance(rf_final_pred, "lift", "rpp")
plot(perf, main='Lift chart')
```




```
# ROC Curve
perf <- performance(rf_final_pred, "tpr", "fpr")
plot(perf, main='ROC chart')
```




```r
# auc
auc <- performance(rf_final_pred, "auc")
auc <- unlist(slot(auc, "y.values"))
print(paste0("ROC AUC Score for the Final Model is:",auc))
```

```
## [1] "ROC AUC Score for the Final Model is:0.86947726169283"
```

CONCLUSION

-The complete metrics and confusion matrix are shown below :

```r
print("Test Confusion Matrix of final model: ")
```

```
## [1] "Test Confusion Matrix of final model: "
```

```r
cm_rf_final
```

```
##    rf_final_class
##       0   1
##   0 110  57
##   1  23 273
```

```r
print("Accuracy, Recall, Precision and F-score :")
```

```
## [1] "Accuracy, Recall, Precision and F-score :"
```

```r
metrics(cm_rf_final)
```

```
## [1] "Test accuracy : 0.827213822894168"
## [1] "Recall of 1 (Success) : 0.922297297297297"
## [1] "Precision of 1 (Success) : 0.827272727272727"
## [1] "f score of 1 (Success) : 0.87220447284345"
```

```r
sp <- cm[1,1]/(cm[1,1]+cm[1,2])
fpr <- cm[1,2]/(cm[2,2]+cm[2,1])
fnr <- cm[2,1]/(cm[1,1]+cm[2,1])
print(paste("Specificity (tnr) :",sp))
```

```
## [1] "Specificity (tnr) : 0.640718562874252"
```

```r
print(paste("False positve rate:",fpr))
```

```
## [1] "False positve rate: 0.202702702702703"
```

```r
print(paste("False negative rate:",fnr))
```

```
## [1] "False negative rate: 0.176923076923077"
```

Conclusions:

- 2 base models (decision tree and Random Forest) were built to compare the performance and to select one as the final model.
- After tuning the parameters, cross validation and plotting the ROC curve for both models, we were able to conclude that Random Forest is performing slightly better than our pruned decision tree. Hence, Random Forest was chosen as the final model
- From the above plot and table we can infer that the important variables are:
  - Is.Non.Annual
  - SingleGradeTripFlag
  - Spr.New.Existing
  - FPP.to.School.enrollment
  - Total.Discount.pax We can also see that our feature extracted variables also have a relatively high importance, hence it was worthwhile to do the extractions.
- The final random forest model had the below performance metrics :
  - Test accuracy : 0.827213822894168
  - Recall of 1 (Success) : 0.922297297297297
  - Precision of 1 (Success) : 0.827272727272727
  - F score of 1 (Success) : 0.87220447284345
- Since we want to reduce the cost and maximize the yield, the models were optimized for it, which is reflected in the results. Final Model has a F-Score of 0.827 which is really good for our business problem, also the Type I and Type II errors are also relatively small, hence even though there's room for some improvement, the built final model should introduce some helpful process to the business.

——————————END——————————