Porto Seguro's Safe Driver Prediction

Shwet Prakash

Problem Statement

Porto Seguro's Insurance challenged Kagglers to build models that calculate the probability that a driver will file a claim in the next year. Hopefully, the models will help lower the cost for good drivers.

Explanation of Case Study

Porto Seguro, one of Brazil's largest auto and homeowner insurance companies wants to avoid the inaccuracies in car insurance company's claim predictions which results in raise the cost of insurance for good drivers and reduce the price for bad ones. The challenge is to build a model that predicts the probability that a driver will initiate an auto insurance claim in the next year.

Data Description

In the train and test data, features that belong to similar groupings are tagged as such in the feature names (e.g., ind, reg, car, calc). In addition, feature names include the postfix bin to indicate binary features and cat to indicate categorical features. Features without these designations are either continuous or ordinal. Values of -1 indicate that the feature was missing from the observation. The target columns signifies whether or not a claim was filed for that policy holder. File Description • train.csv contains the training data, where each row corresponds to a policy holder, and the target columns signifies that a claim was filed. • test.csv contains the test data. • sample_submission.csv is submission file showing the correct format.

Loading Required Libraries

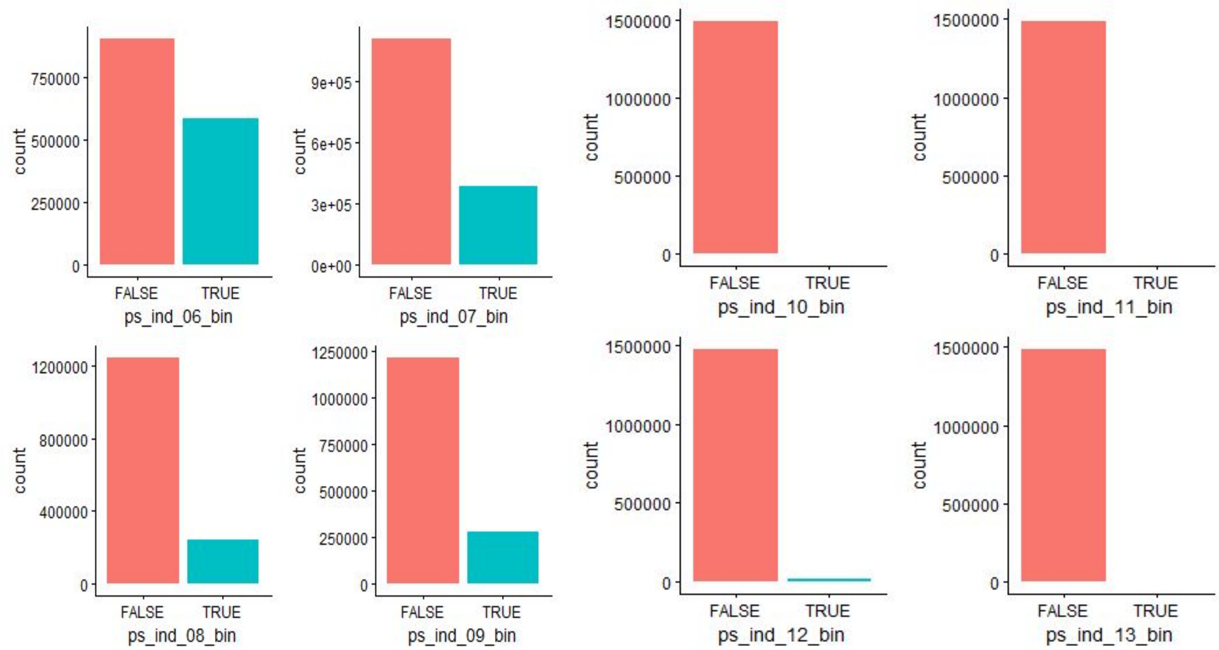
I used the following libraries in the project:

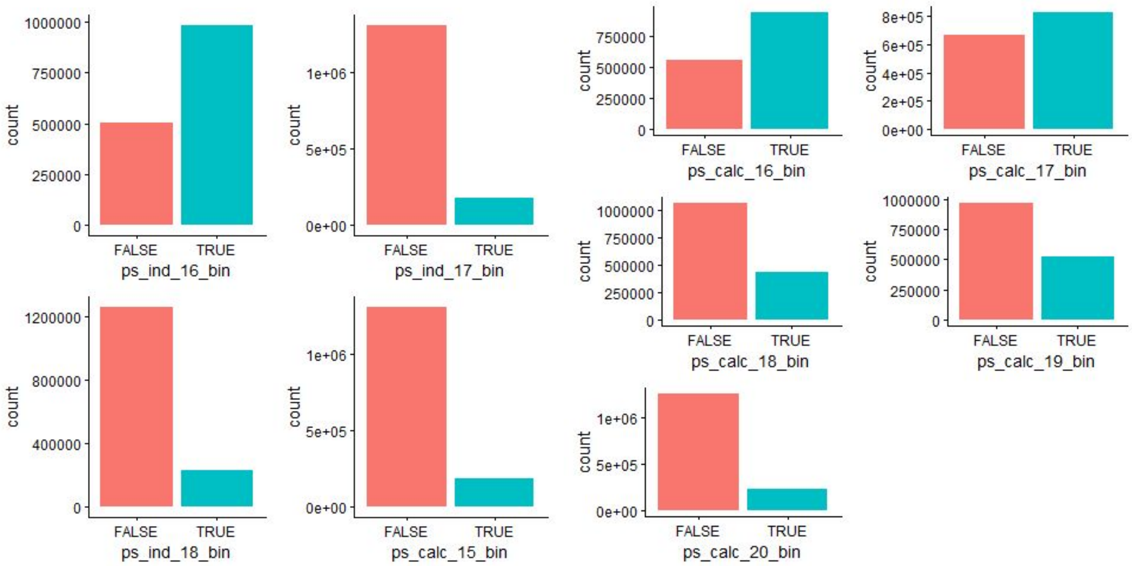
```
library(dplyr) #data manipulation
library(readr) #input/output
library(data.table) #data manipulation
library(stringr) #string manipulation
library(caret)  #model evaluation (confusion matrix)
library(tibble) #data wrangling
library("ROSE") #over/under sampling
library("randomForest") #random forest model building
```

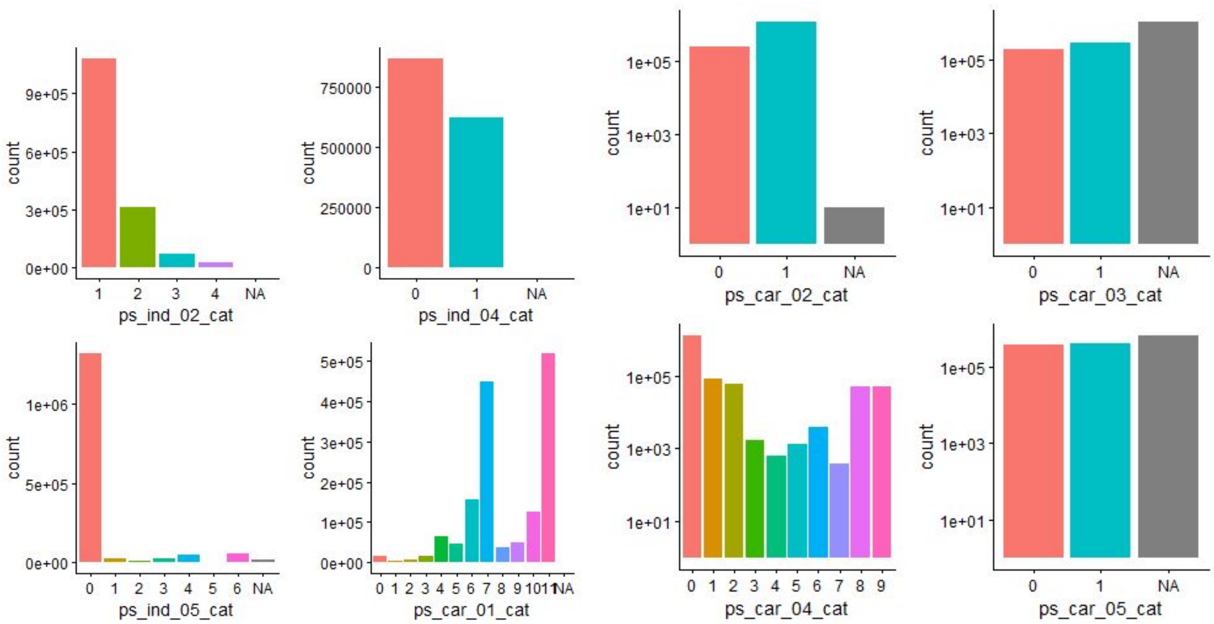
```
library(pROC) #ROC plots
library("MLmetrics") #Normalized Gini
library(corrplot) #finding correlation
library(cowplot) #simple add on to ggplot2
library(xgboost)
library(verification)
library(Matrix)
```

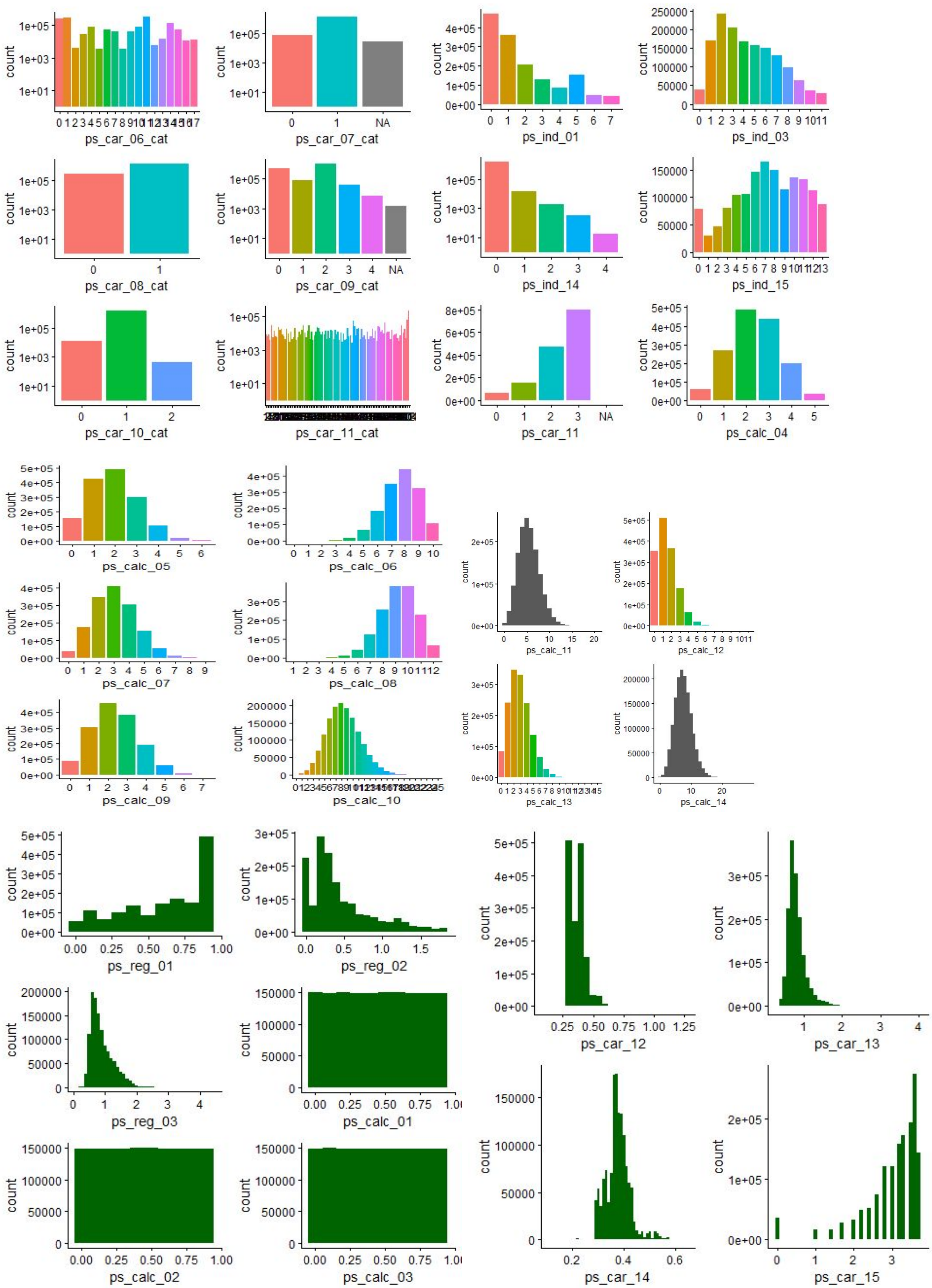
Exploratory Data Analysis(Data Visualization)

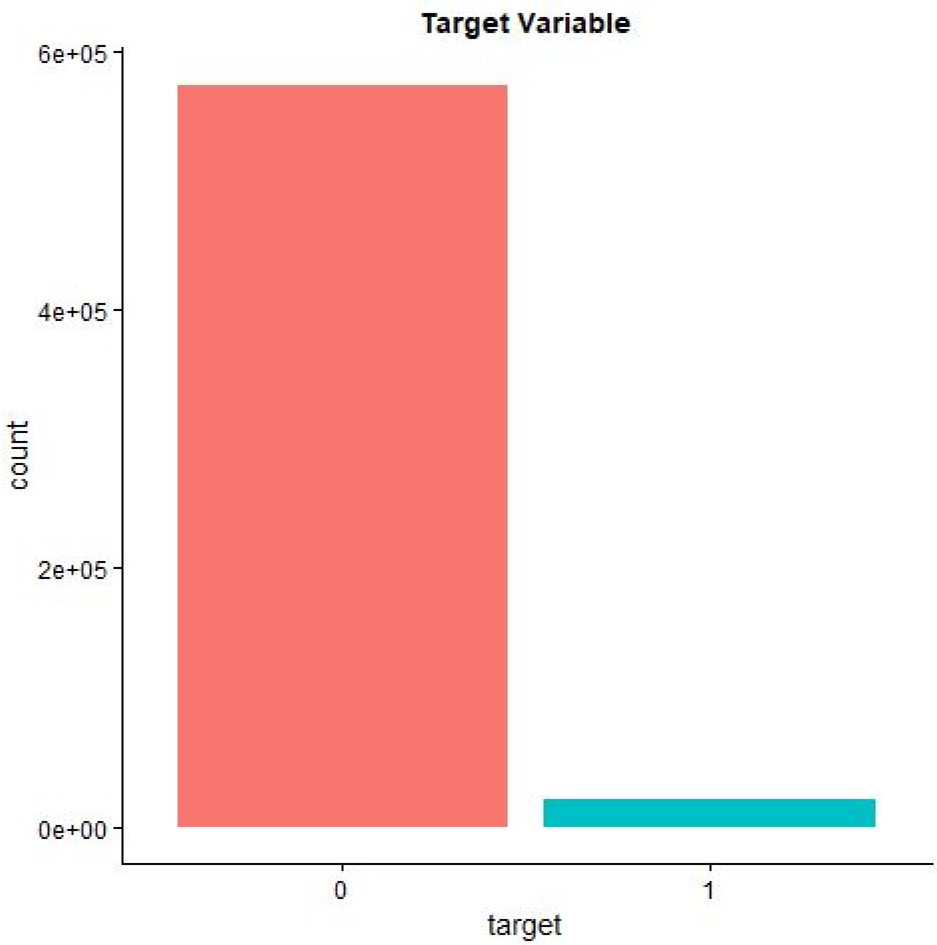
1. **Individual feature visualization**



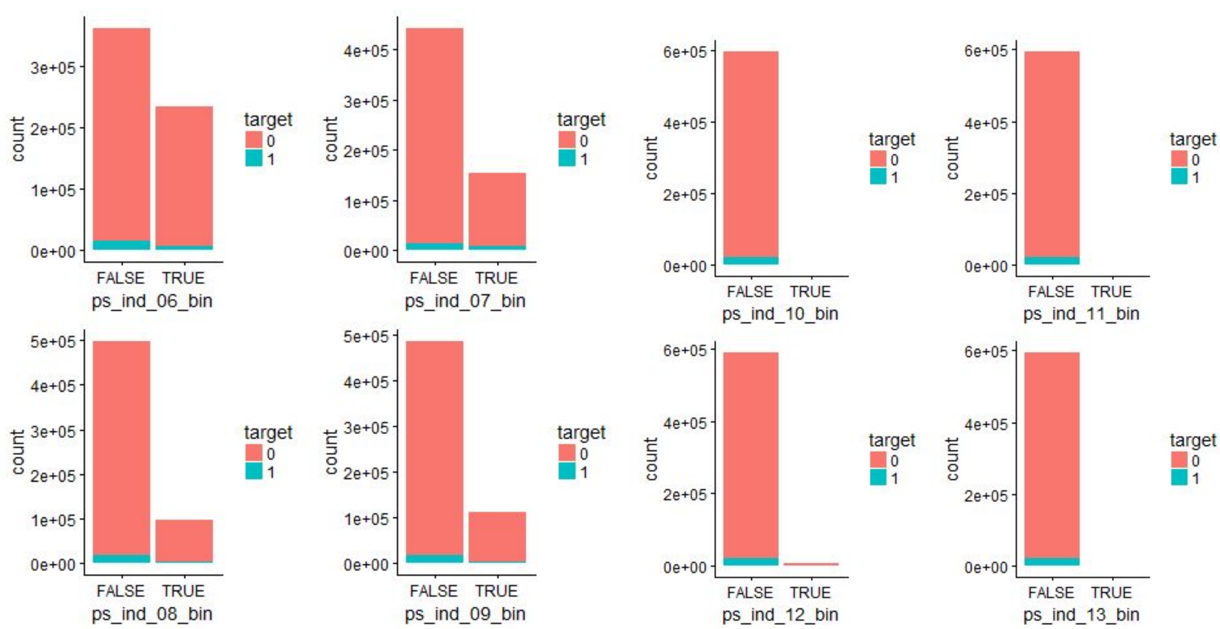

count
FALSE
TRUE
ps_ind_16_bin
ps_ind_17_bin
ps_calc_16_bin
ps_calc_17_bin
ps_calc_18_bin
ps_calc_19_bin
ps_ind_18_bin
ps_calc_15_bin
ps_calc_20_bin

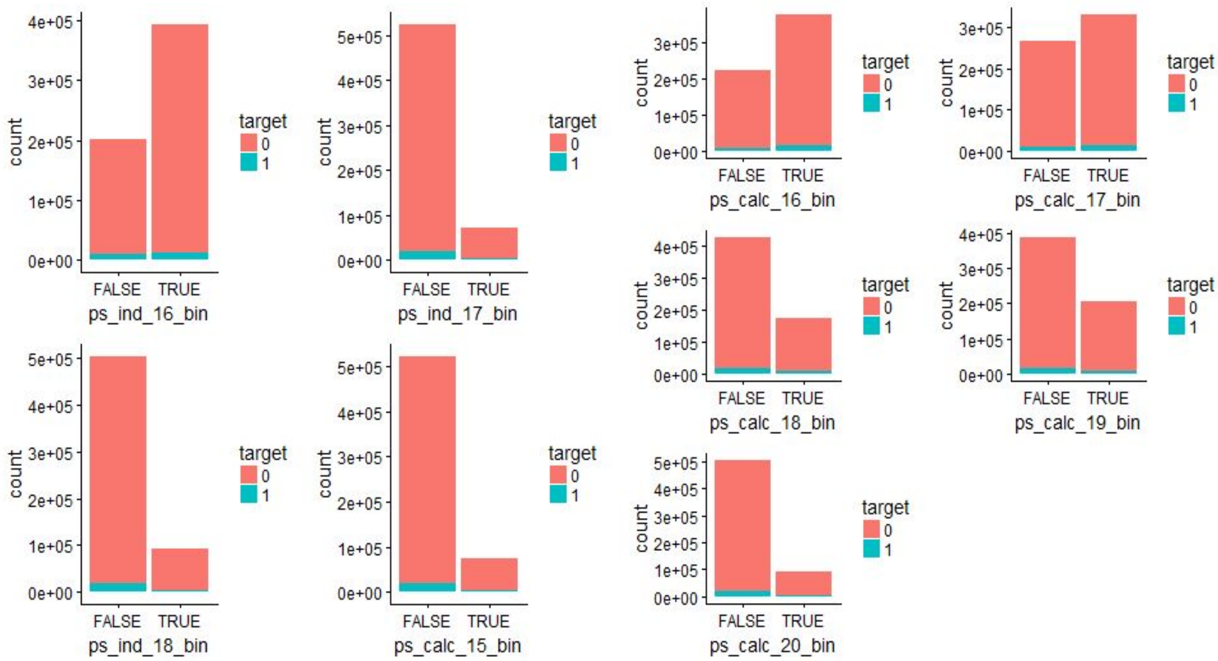

count
ps_ind_02_cat
ps_ind_04_cat
ps_car_02_cat
ps_car_03_cat
ps_ind_05_cat
ps_car_01_cat
ps_car_04_cat
ps_car_05_cat

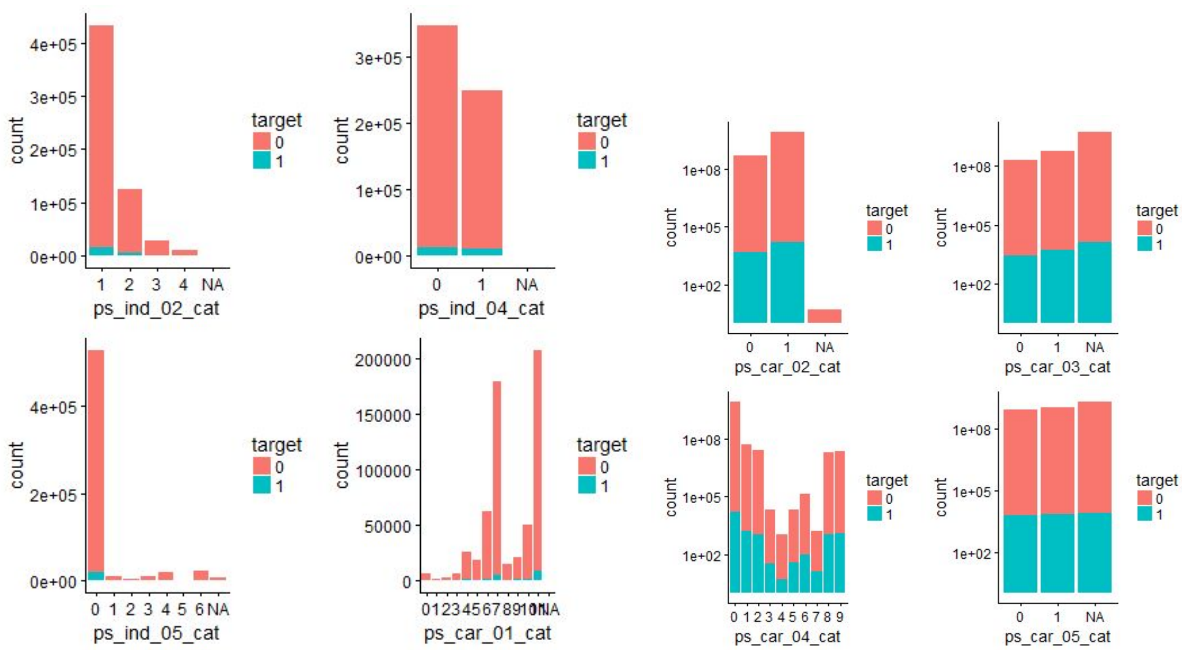

count
ps_car_06_cat
ps_car_07_cat
ps_ind_01
ps_ind_03
ps_car_08_cat
ps_car_09_cat
ps_ind_14
ps_ind_15
ps_car_10_cat
ps_car_11_cat
ps_car_11
ps_calc_04
ps_calc_05
ps_calc_06
ps_calc_11
ps_calc_12
ps_calc_07
ps_calc_08
ps_calc_09
ps_calc_10
ps_calc_13
ps_calc_14
ps_reg_01
ps_reg_02
ps_car_12
ps_car_13
ps_reg_03
ps_calc_01
ps_calc_02
ps_calc_03
ps_car_14
ps_car_15

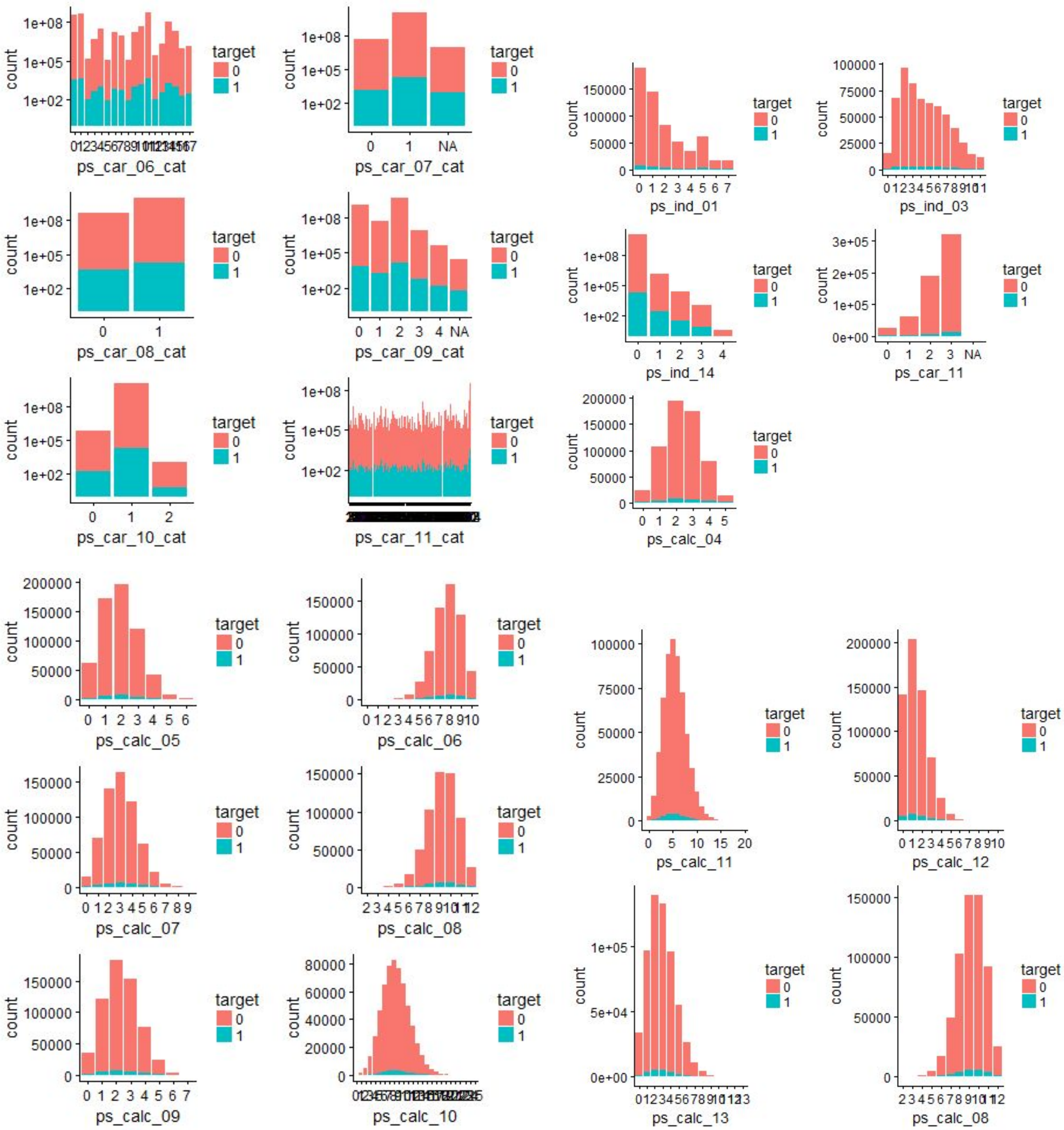


2. Relationship between each feature with the target variable

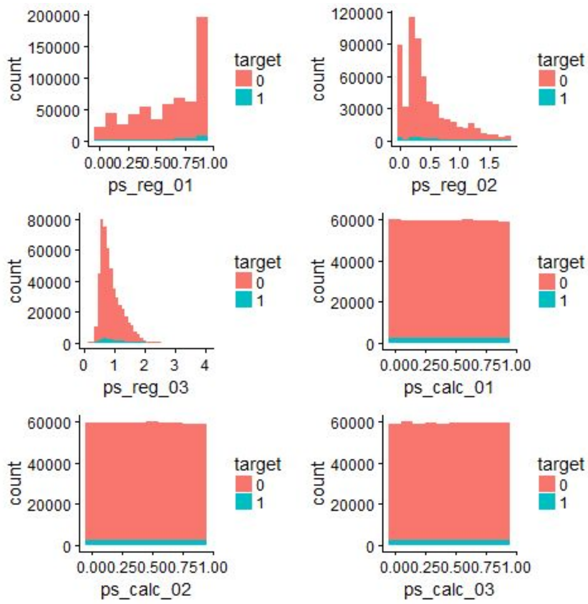


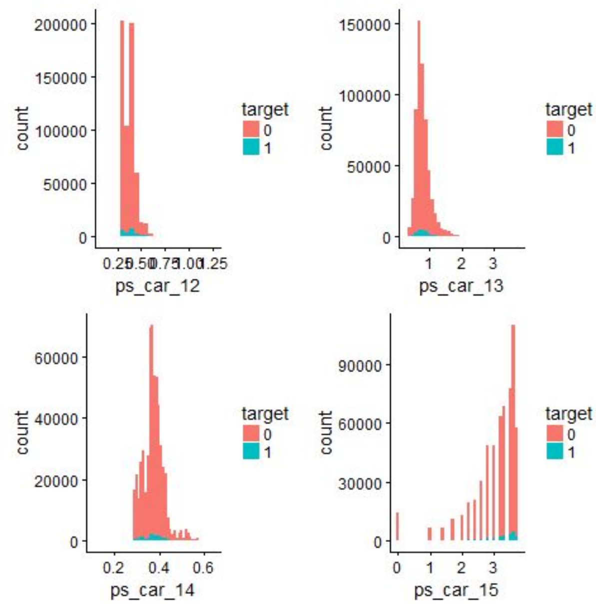

count
target
0
1
FALSE TRUE
ps_ind_16_bin
ps_ind_17_bin
ps_calc_16_bin
ps_calc_17_bin
ps_calc_18_bin
ps_calc_19_bin
ps_ind_18_bin
ps_calc_15_bin
ps_calc_20_bin
0e+00
1e+05
2e+05
3e+05
4e+05
5e+05


count
target
0
1
ps_ind_02_cat
ps_ind_04_cat
ps_car_02_cat
ps_car_03_cat
ps_ind_05_cat
ps_car_01_cat
ps_car_04_cat
ps_car_05_cat
0e+00
1e+05
2e+05
3e+05
4e+05
0
50000
100000
150000
200000
1e+02
1e+05
1e+08
NA

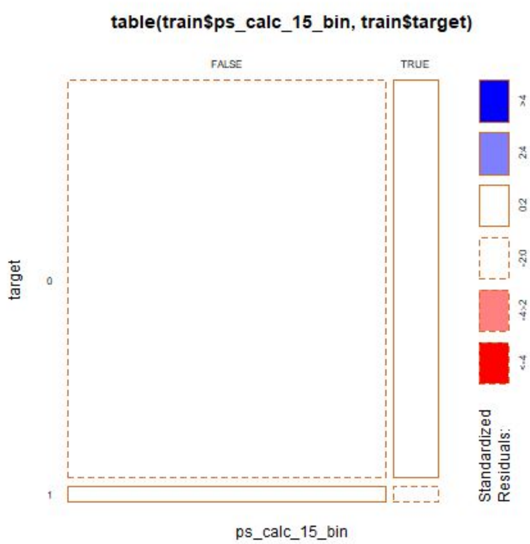


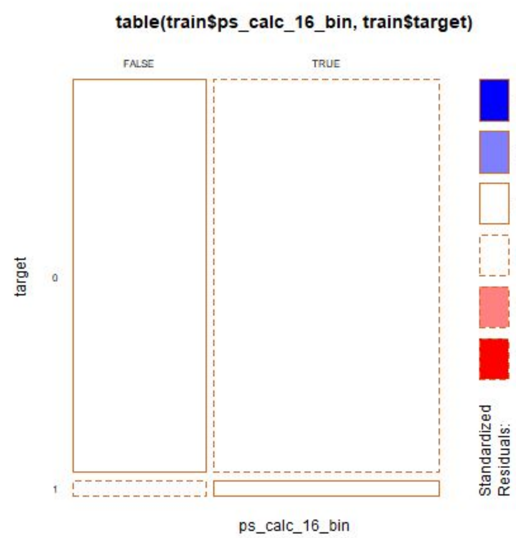
ps_car_06_cat
ps_car_07_cat
ps_ind_01
ps_ind_03
ps_car_08_cat
ps_car_09_cat
ps_ind_14
ps_car_11
ps_car_10_cat
ps_car_11_cat
ps_calc_04
ps_calc_05
ps_calc_06
ps_calc_11
ps_calc_12
ps_calc_07
ps_calc_08
ps_calc_09
ps_calc_10
ps_calc_13
ps_calc_08
count
target
0
1

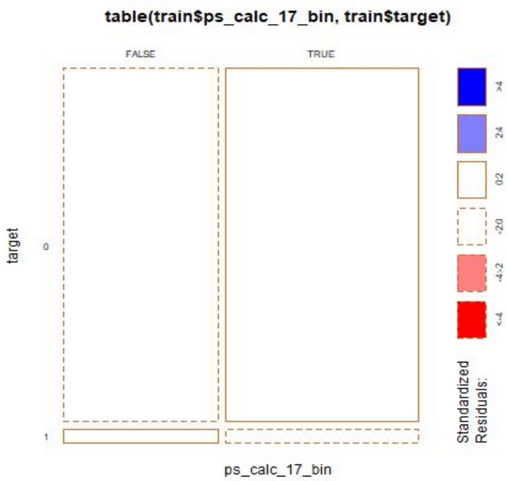


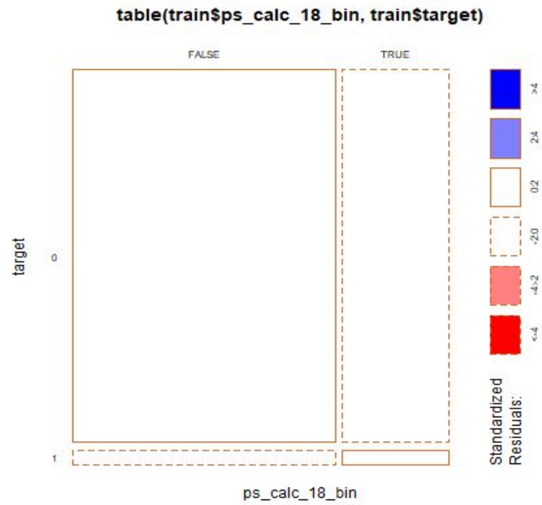


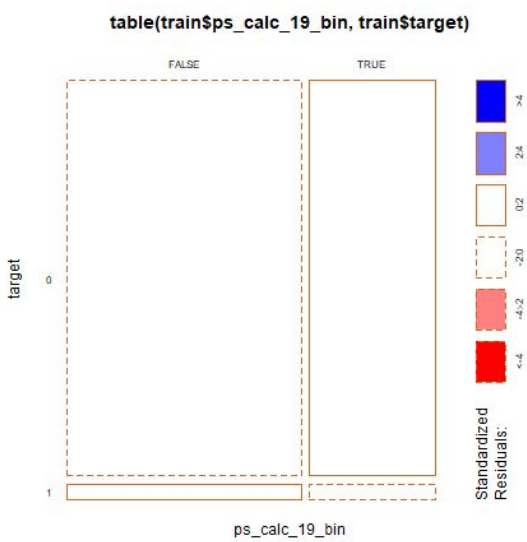
3. Mosaic plots

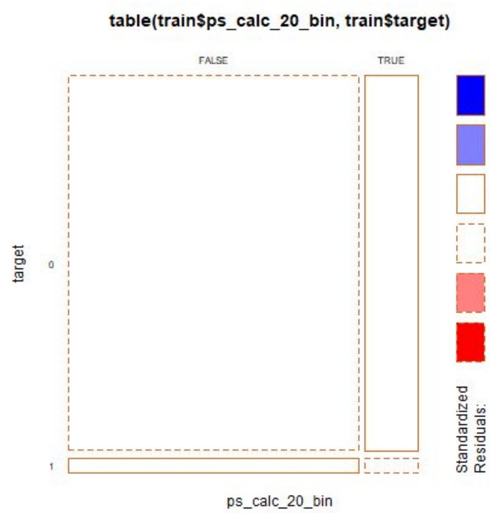


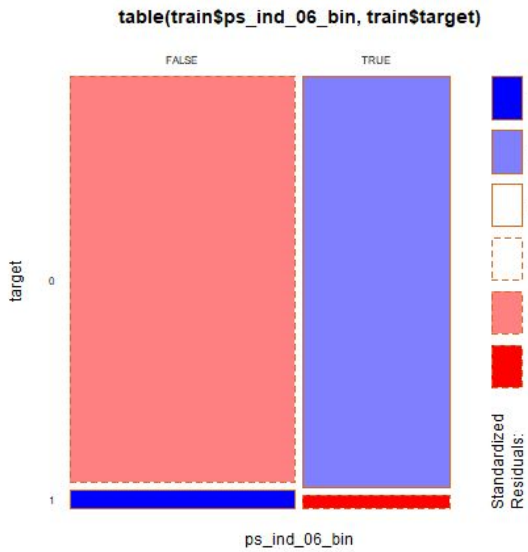


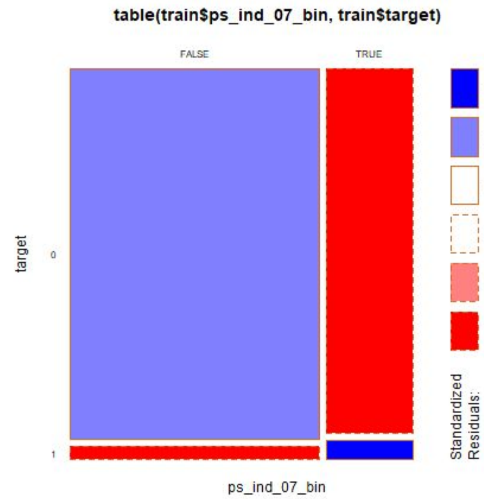

table(train$ps_calc_17_bin, train$target)
FALSE
TRUE
target
0
1
ps_calc_17_bin
Standardized Residuals:

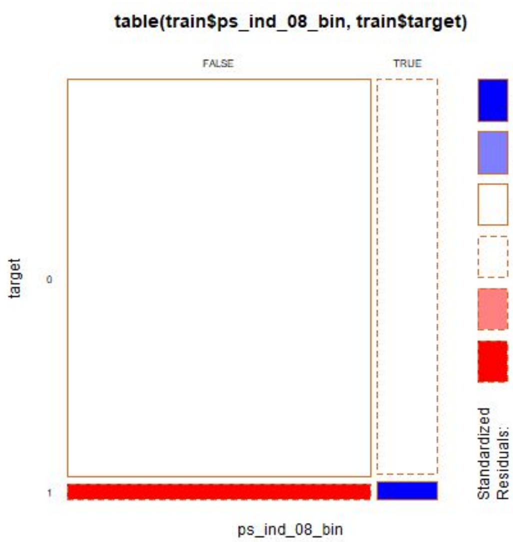

table(train$ps_calc_18_bin, train$target)
FALSE
TRUE
target
0
1
ps_calc_18_bin
Standardized Residuals:

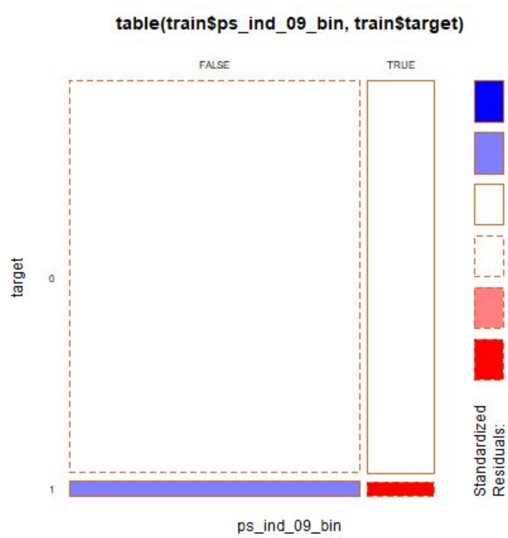

table(train$ps_calc_19_bin, train$target)
FALSE
TRUE
target
0
1
ps_calc_19_bin
Standardized Residuals:

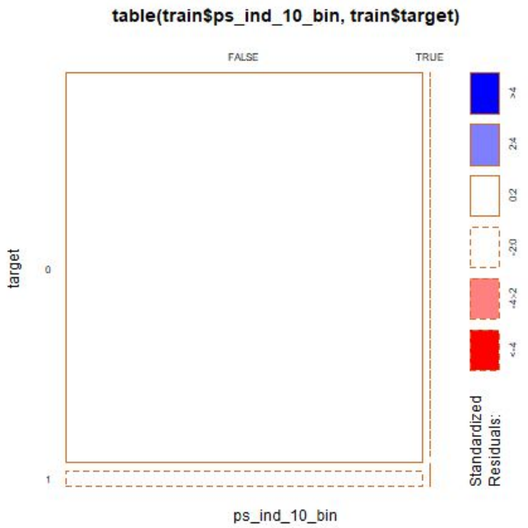

table(train$ps_calc_20_bin, train$target)
FALSE
TRUE
target
0
1
ps_calc_20_bin
Standardized Residuals:

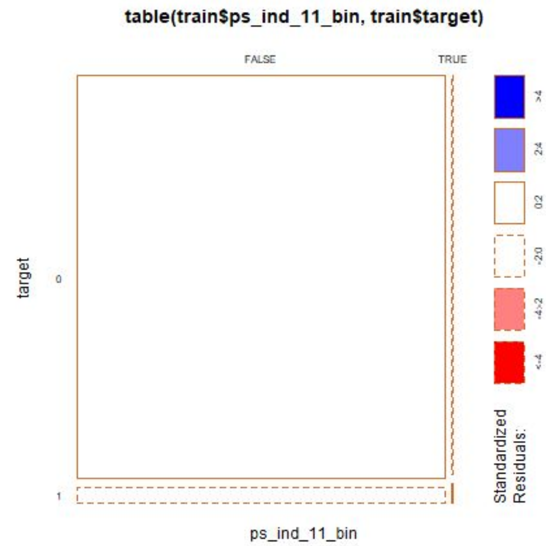

table(train$ps_ind_06_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_06_bin
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

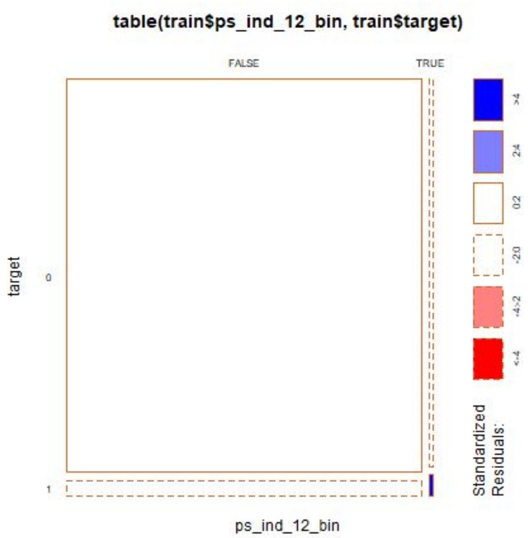

table(train$ps_ind_07_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_07_bin
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

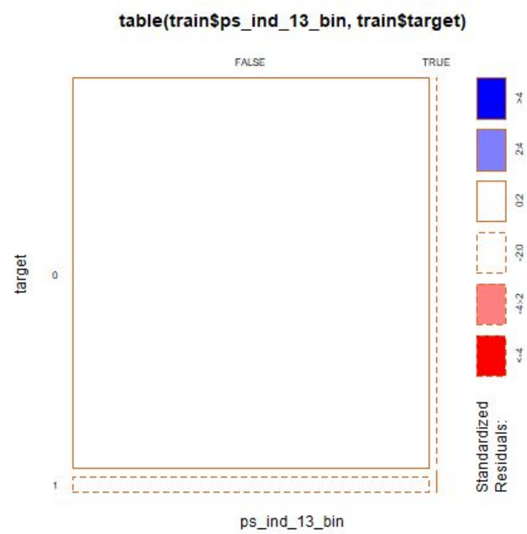

table(train$ps_ind_08_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_08_bin
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

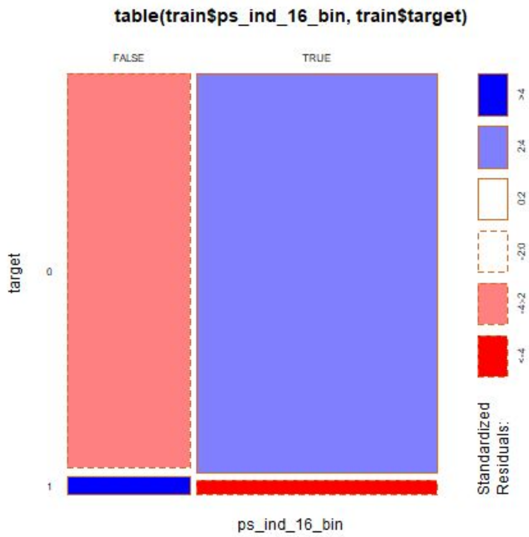

table(train$ps_ind_09_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_09_bin
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

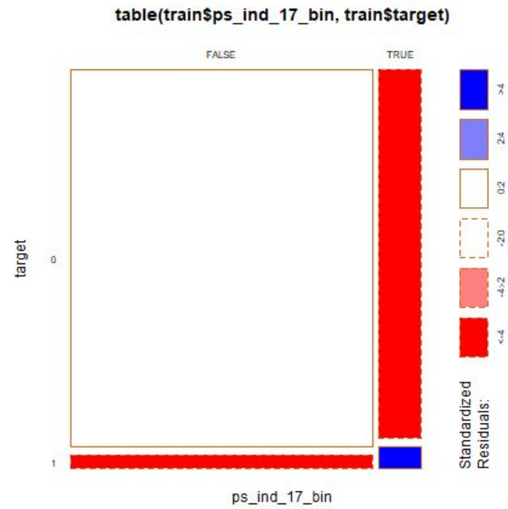

table(train$ps_ind_10_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_10_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized Residuals:

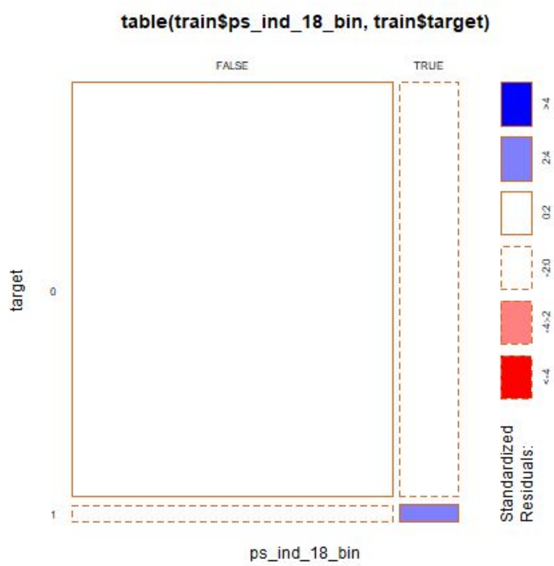

table(train$ps_ind_11_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_11_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized Residuals:

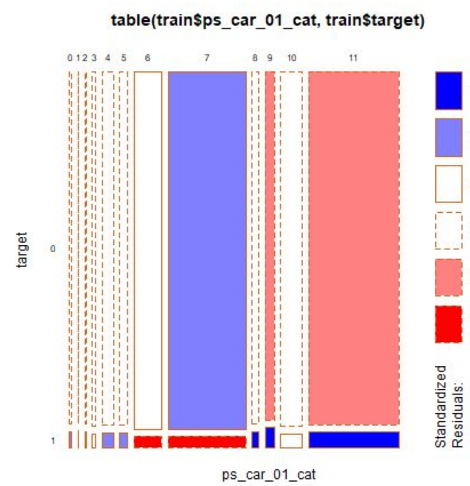

table(train$ps_ind_12_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_12_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized Residuals:

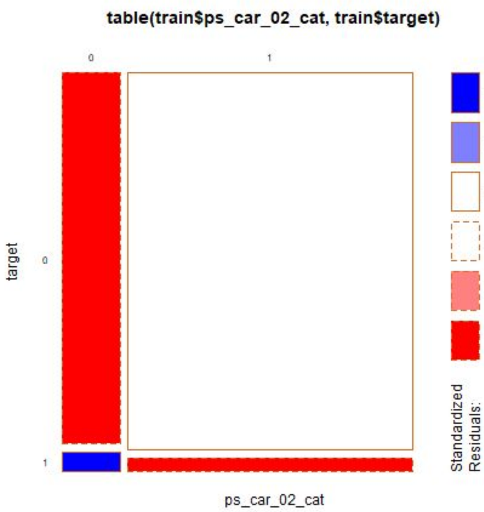

table(train$ps_ind_13_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_13_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized Residuals:

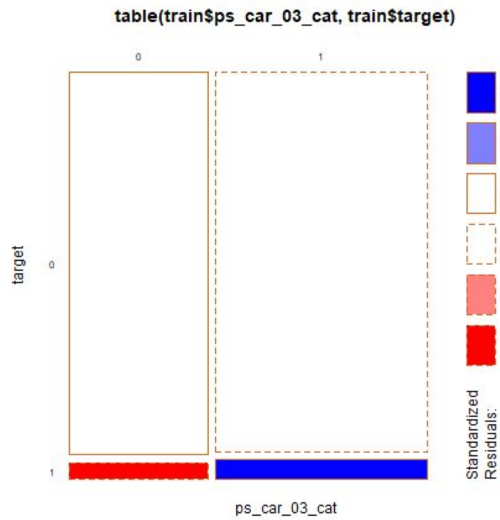

table(train$ps_ind_16_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_16_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

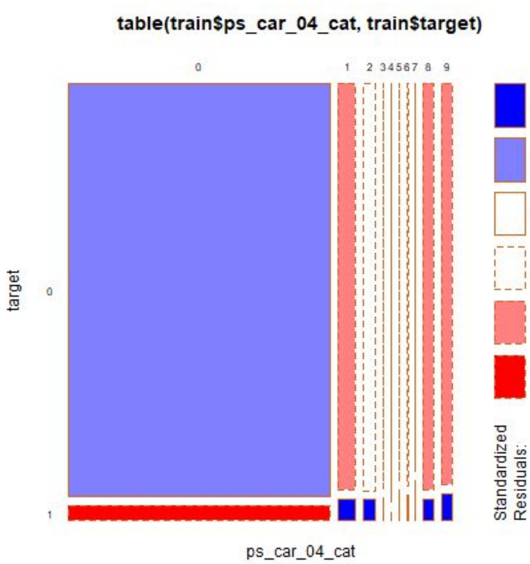

table(train$ps_ind_17_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_17_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

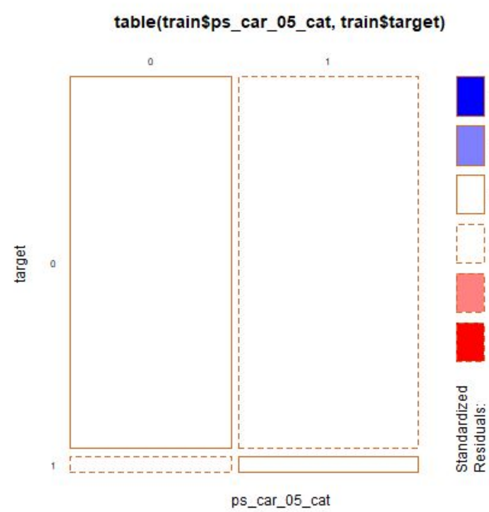

table(train$ps_ind_18_bin, train$target)
FALSE
TRUE
target
0
1
ps_ind_18_bin
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

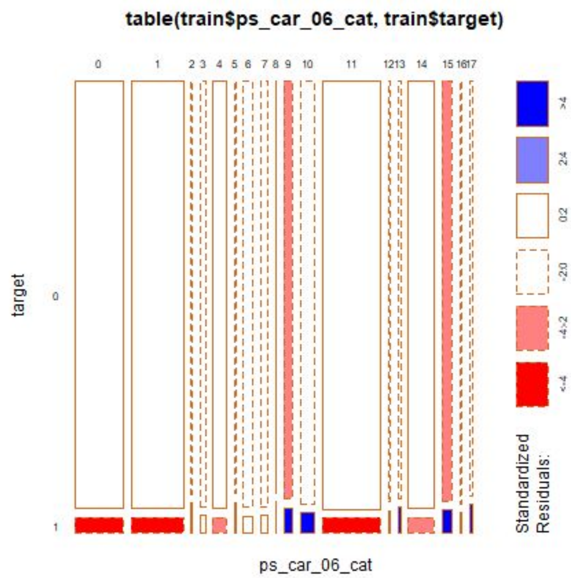

table(train$ps_car_01_cat, train$target)
0 1 2 3 4 5 6 7 8 9 10 11
target
0
1
ps_car_01_cat
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

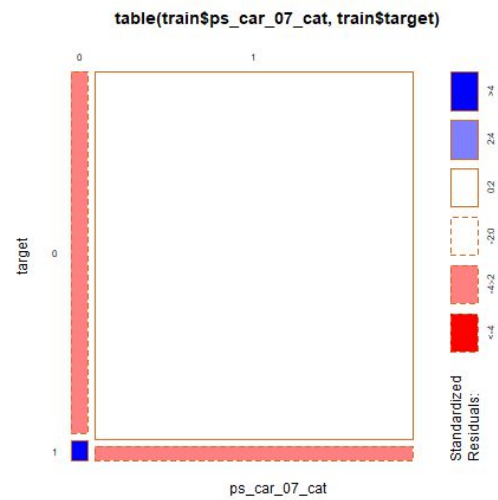

table(train$ps_car_02_cat, train$target)
target
ps_car_02_cat
Standardized Residuals:

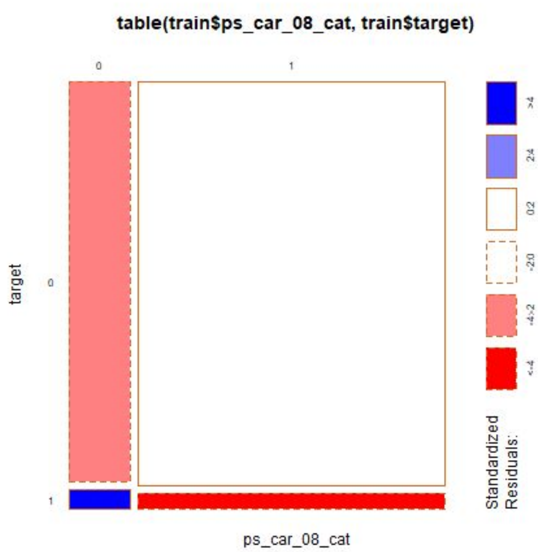

table(train$ps_car_03_cat, train$target)
target
ps_car_03_cat
Standardized Residuals:

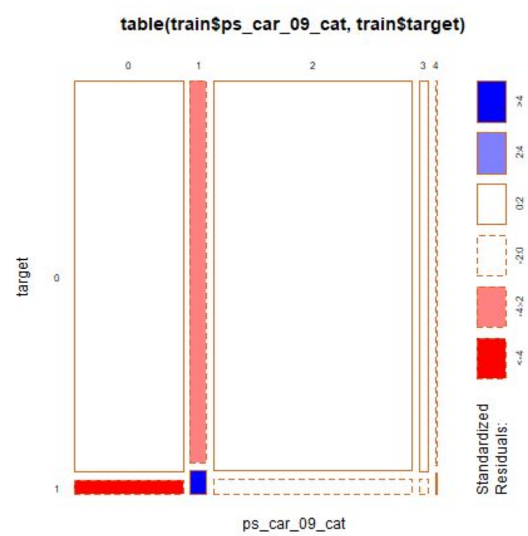

table(train$ps_car_04_cat, train$target)
target
ps_car_04_cat
Standardized Residuals:

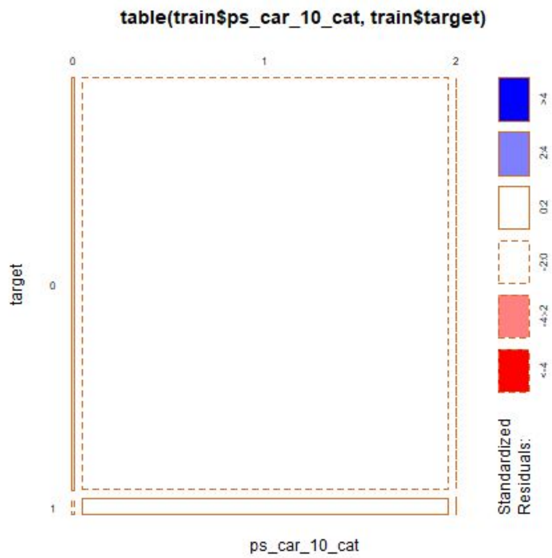

table(train$ps_car_05_cat, train$target)
target
ps_car_05_cat
Standardized Residuals:

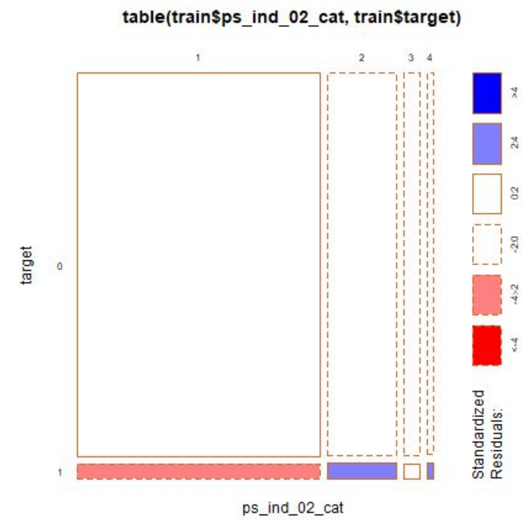

table(train$ps_car_06_cat, train$target)
0 1 2 3 4 5 6 7 8 9 10 11 1213 14 15 1617
target
0
1
ps_car_06_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

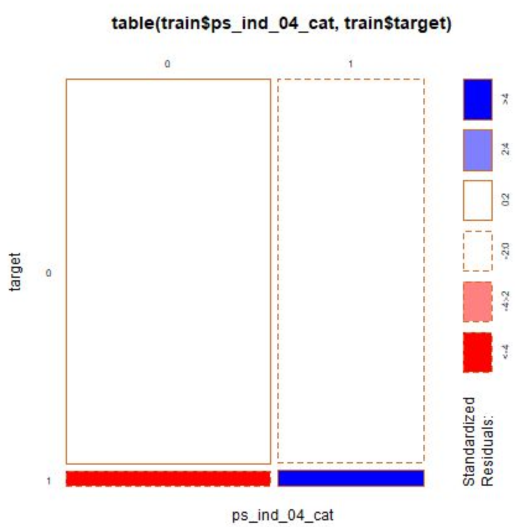

table(train$ps_car_07_cat, train$target)
0 1
target
0
1
ps_car_07_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

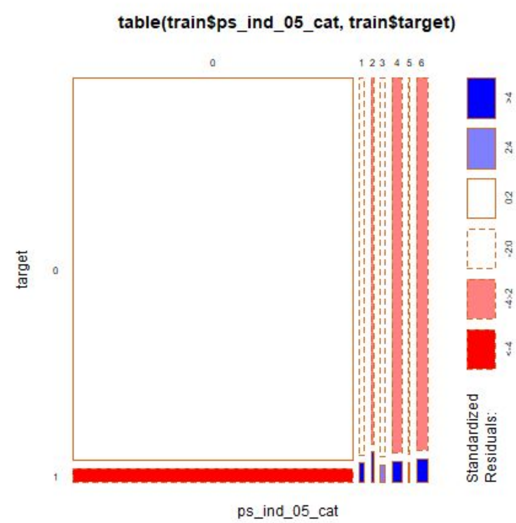

table(train$ps_car_08_cat, train$target)
0 1
target
0
1
ps_car_08_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

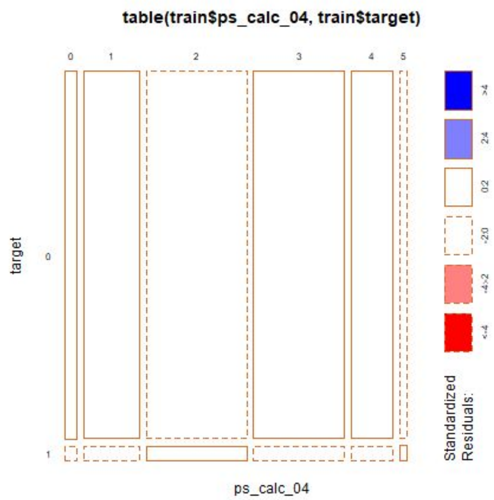

table(train$ps_car_09_cat, train$target)
0 1 2 3 4
target
0
1
ps_car_09_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

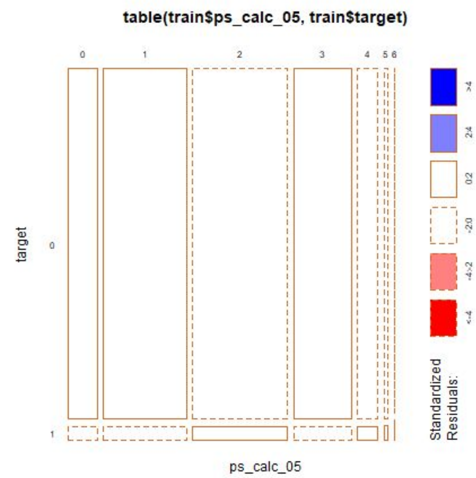

table(train$ps_car_10_cat, train$target)
0
1
2
target
0
1
ps_car_10_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

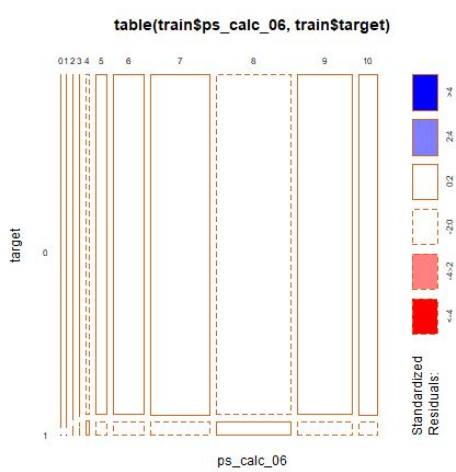

table(train$ps_ind_02_cat, train$target)
1
2
3
4
target
0
1
ps_ind_02_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

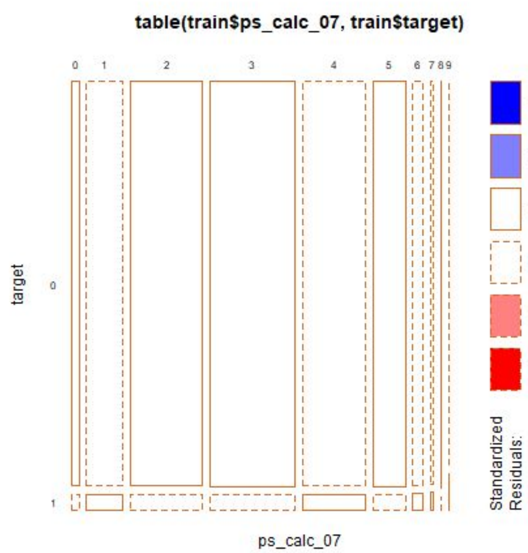

table(train$ps_ind_04_cat, train$target)
0
1
target
0
1
ps_ind_04_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

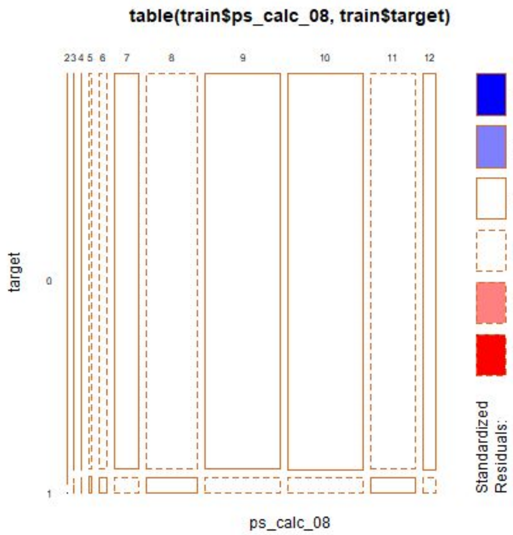

table(train$ps_ind_05_cat, train$target)
0
1
2
3
4
5
6
target
0
1
ps_ind_05_cat
Standardized Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

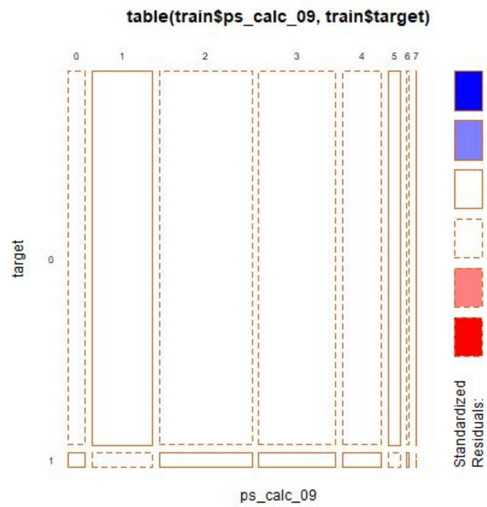

table(train$ps_calc_04, train$target)
0 1 2 3 4 5
target
0
1
ps_calc_04
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

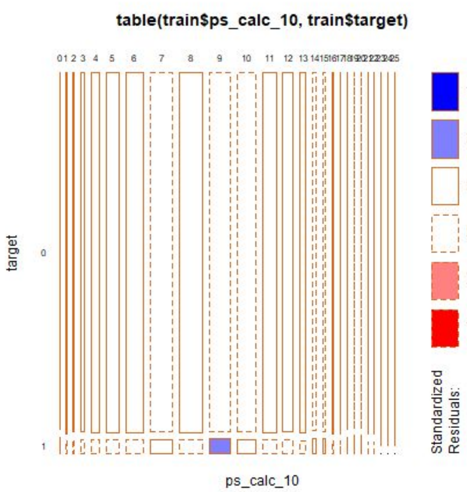

table(train$ps_calc_05, train$target)
0 1 2 3 4 5 6
target
0
1
ps_calc_05
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

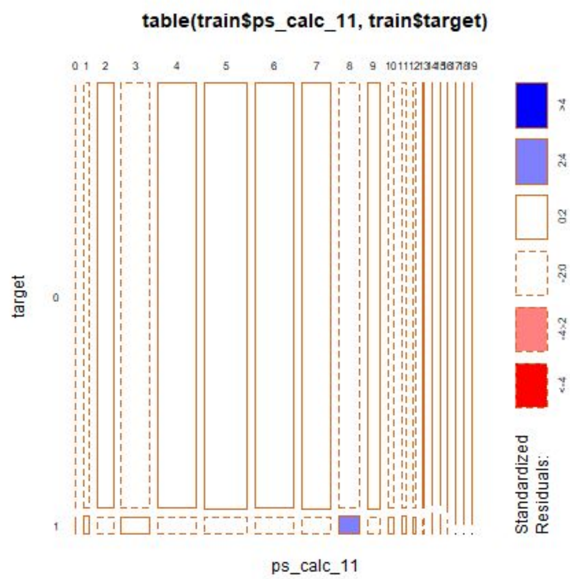

table(train$ps_calc_06, train$target)
0 1 2 3 4 5 6 7 8 9 10
target
0
1
ps_calc_06
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

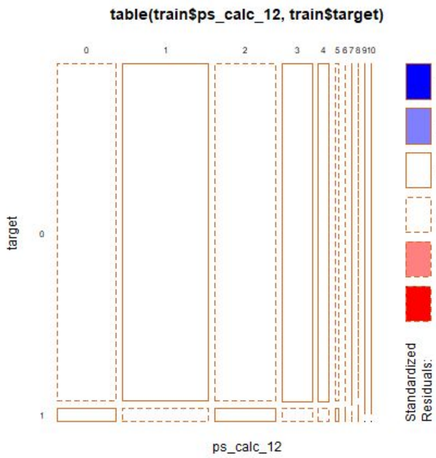

table(train$ps_calc_07, train$target)
0 1 2 3 4 5 6 7 8 9
target
0
1
ps_calc_07
>4
2:4
0:2
-2:0
-4:-2
<-4
Standardized
Residuals:

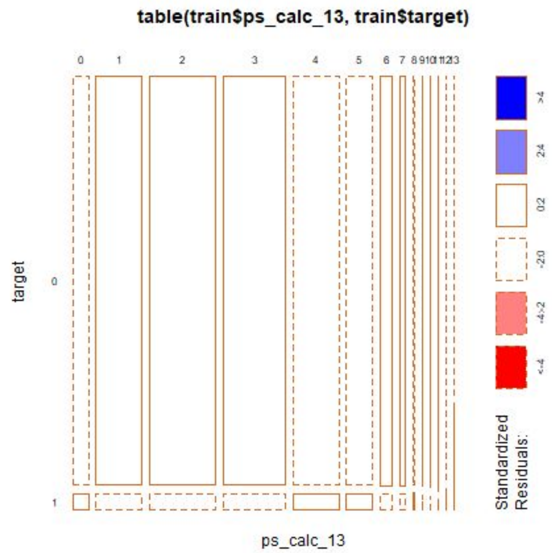

table(train$ps_calc_08, train$target)
target
ps_calc_08
Standardized Residuals:

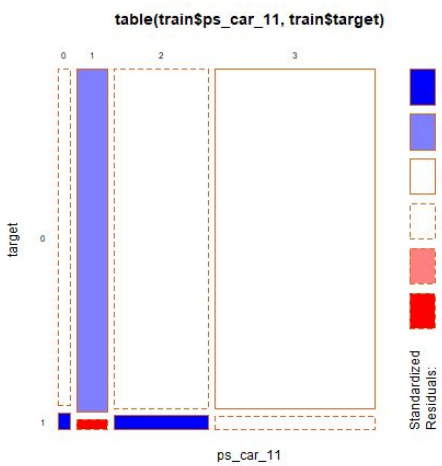

table(train$ps_calc_09, train$target)
target
ps_calc_09
Standardized Residuals:

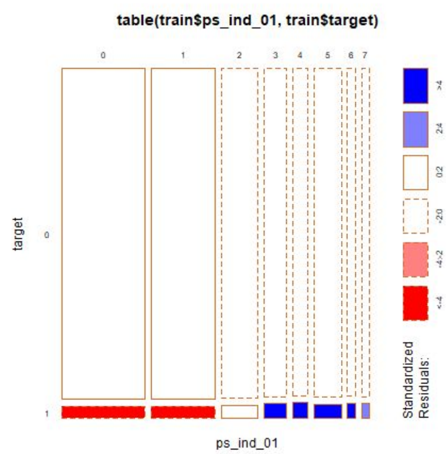

table(train$ps_calc_10, train$target)
target
ps_calc_10
Standardized Residuals:

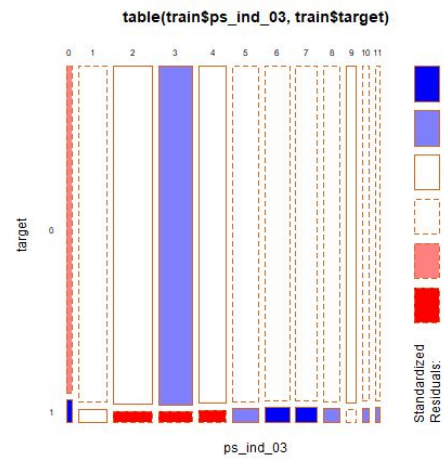

table(train$ps_calc_11, train$target)
target
ps_calc_11
Standardized Residuals:

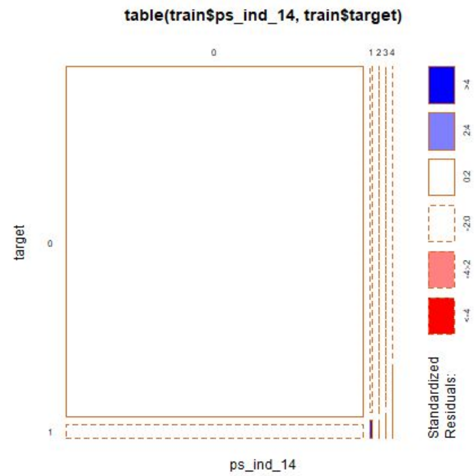

table(train$ps_calc_12, train$target)
0
1
2
3
4
5678910
target
0
1
ps_calc_12
Standardized
Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

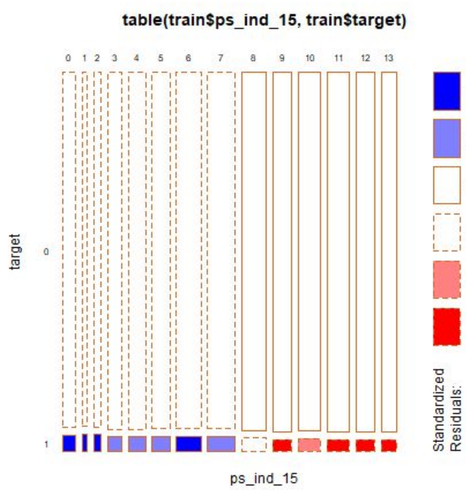

table(train$ps_calc_13, train$target)
0
1
2
3
4
5
6
7
8910111213
target
0
1
ps_calc_13
Standardized
Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

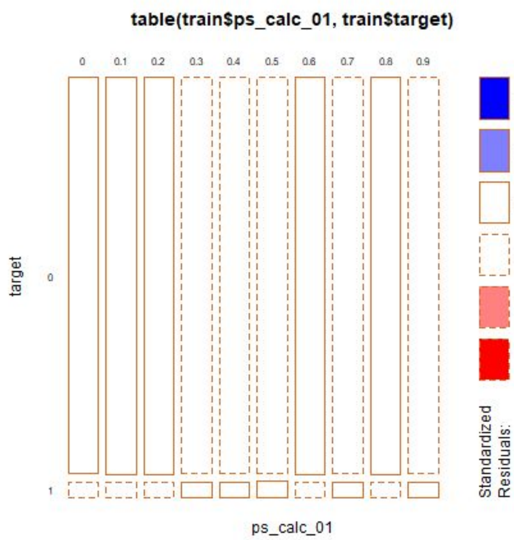

table(train$ps_car_11, train$target)
0
1
2
3
target
0
1
ps_car_11
Standardized
Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

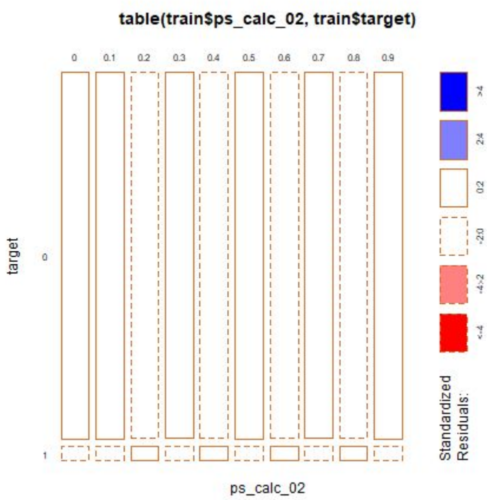

table(train$ps_ind_01, train$target)
0
1
2
3
4
5
6
7
target
0
1
ps_ind_01
Standardized
Residuals:
>4
2:4
0:2
-2:0
-4:-2
<-4

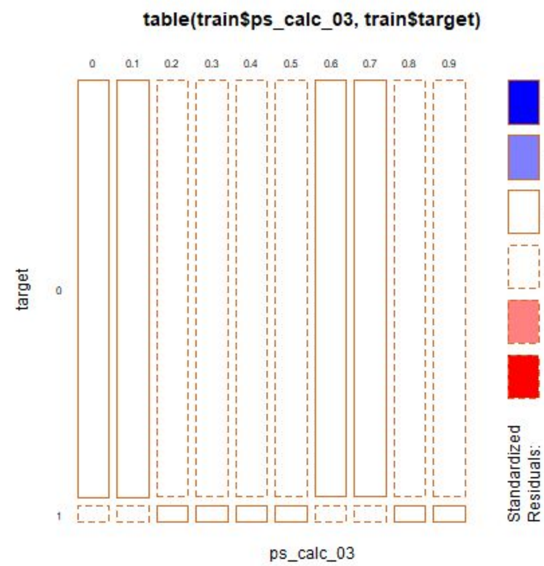

table(train$ps_ind_03, train$target)
0 1 2 3 4 5 6 7 8 9 10 11
target
0
1
ps_ind_03
Standardized Residuals:
<-4
-4:-2
-2:0
0:2
2:4
>4

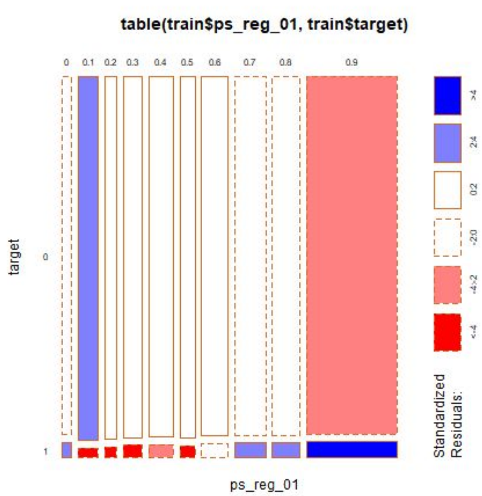

table(train$ps_ind_14, train$target)
0 1 2 3 4
target
0
1
ps_ind_14
Standardized Residuals:
<-4
-4:-2
-2:0
0:2
2:4
>4

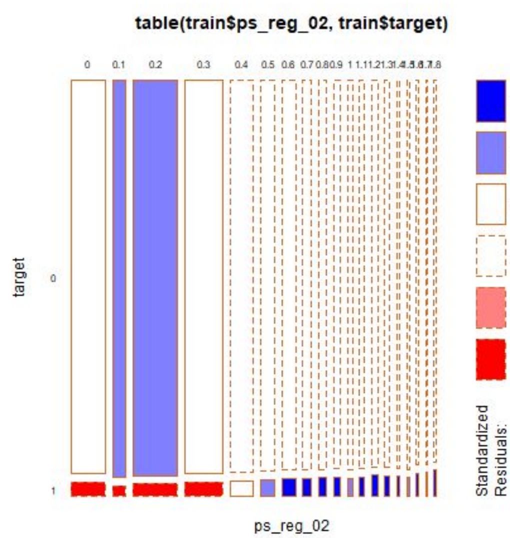

table(train$ps_ind_15, train$target)
0 1 2 3 4 5 6 7 8 9 10 11 12 13
target
0
1
ps_ind_15
Standardized Residuals:
<-4
-4:-2
-2:0
0:2
2:4
>4


table(train$ps_calc_01, train$target)
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
target
0
1
ps_calc_01
Standardized Residuals:
<-4
-4:-2
-2:0
0:2
2:4
>4

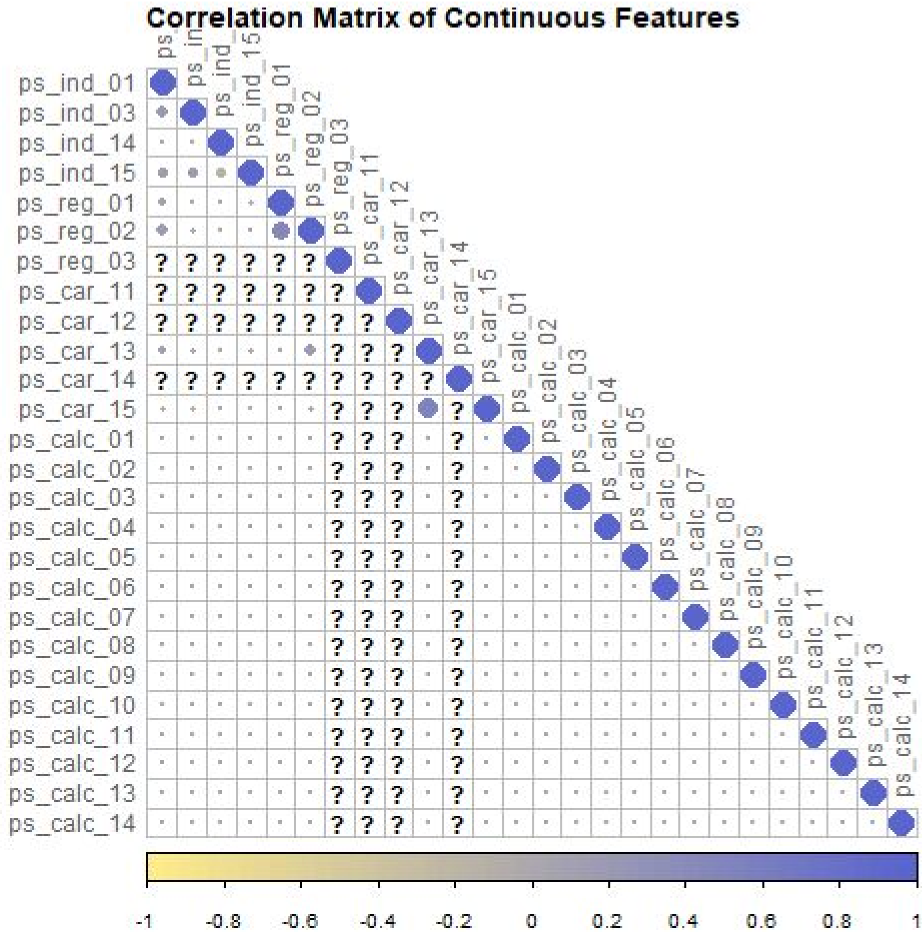

table(train$ps_calc_02, train$target)
target
ps_calc_02
Standardized Residuals:

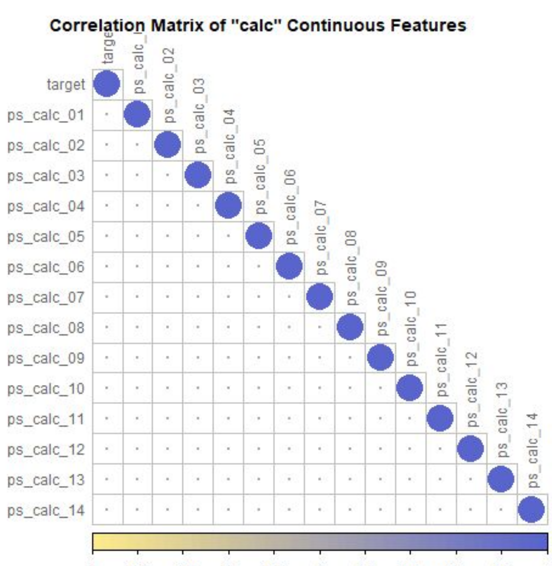

table(train$ps_calc_03, train$target)
target
ps_calc_03
Standardized Residuals:

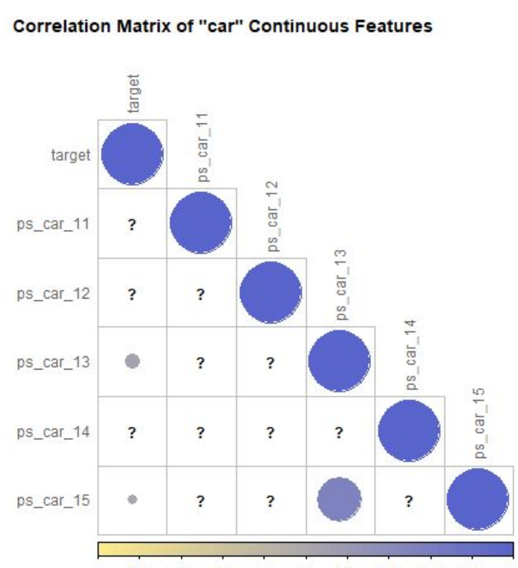

table(train$ps_reg_01, train$target)
target
ps_reg_01
Standardized Residuals:


table(train$ps_reg_02, train$target)
target
ps_reg_02
Standardized Residuals:

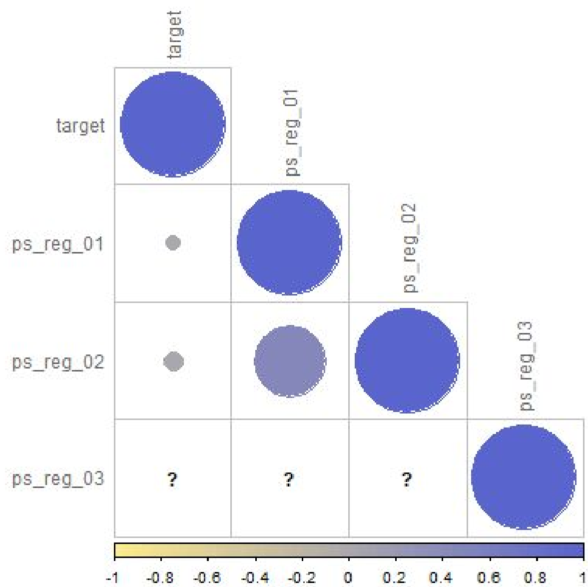
4. Correlation between continuous variables



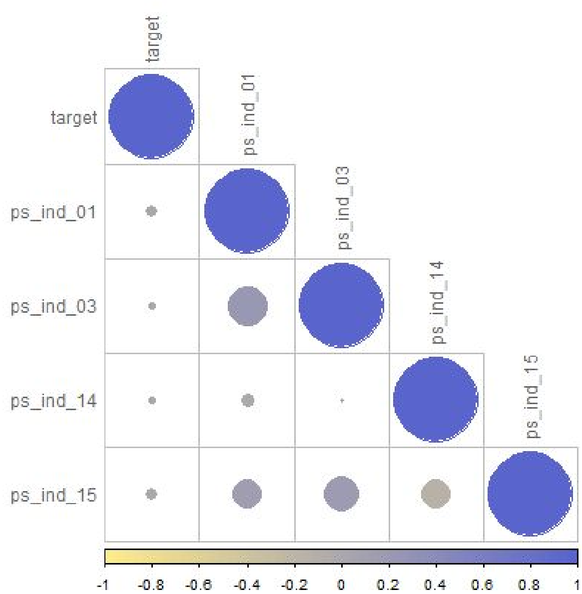




Correlation Matrix of "reg" Continuous Features



Correlation Matrix of "ind" Continuous Features



Data Preprocessing

According to the data description given, values “-1” indicate the features are missing from the observation. So, while importing the data I have considered “-1”,“-1.0” as NAs.

```
## [1] 595212 59
## [1] 892816 58
```

The TRAIN dataset contains 595212 Observations and 59 Variables (including target variable). The TEST dataset contains 892816 Observations and 58 Variables (excluding target variable).

Let’s look at the structure and missing values of the datasets.

str(train)

Classes ‘tbl_df’, ‘tbl’ and 'data.frame': 595212 obs. of 59 variables:

```
 $ id          : int  7 9 13 16 17 19 20 22 26 28 ...
 $ target      : int  0 0 0 0 0 0 0 0 0 1 ...
 $ ps_ind_01    : int  2 1 5 0 0 5 2 5 5 1 ...
 $ ps_ind_02_cat : int  2 1 4 1 2 1 1 1 1 1 ...
 $ ps_ind_03    : int  5 7 9 2 0 4 3 4 3 2 ...
 $ ps_ind_04_cat : int  1 0 1 0 1 0 1 0 1 0 ...
 $ ps_ind_05_cat : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_06_bin : int  0 0 0 1 1 0 0 1 0 0 ...
 $ ps_ind_07_bin : int  1 0 0 0 0 0 1 0 0 1 ...
 $ ps_ind_08_bin : int  0 1 1 0 0 0 0 0 1 0 ...
 $ ps_ind_09_bin : int  0 0 0 0 0 1 0 0 0 0 ...
 $ ps_ind_10_bin : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_11_bin : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_12_bin : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_13_bin : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_14    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_15    : int  11 3 12 8 9 6 8 13 6 4 ...
 $ ps_ind_16_bin : int  0 0 1 1 1 1 1 1 1 0 ...
 $ ps_ind_17_bin : int  1 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_18_bin : int  0 1 0 0 0 0 0 0 0 1 ...
 $ ps_reg_01    : num  0.7 0.8 0 0.9 0.7 0.9 0.6 0.7 0.9 0.9 ...
 $ ps_reg_02    : num  0.2 0.4 0 0.2 0.6 1.8 0.1 0.4 0.7 1.4 ...
 $ ps_reg_03    : num  0.718 0.766 NA 0.581 0.841 ...
 $ ps_car_01_cat : int  10 11 7 7 11 10 6 11 10 11 ...
 $ ps_car_02_cat : int  1 1 1 1 1 0 1 1 1 0 ...
 $ ps_car_03_cat : int  NA NA NA 0 NA NA NA 0 NA 0 ...
 $ ps_car_04_cat : int  0 0 0 0 0 0 0 0 0 1 ...
 $ ps_car_05_cat : int  1 NA NA 1 NA 0 1 0 1 0 ...
 $ ps_car_06_cat : int  4 11 14 11 14 14 11 11 14 14 ...
 $ ps_car_07_cat : int  1 1 1 1 1 1 1 1 1 1 ...
 $ ps_car_08_cat : int  0 1 1 1 1 1 1 1 1 1 ...
 $ ps_car_09_cat : int  0 2 2 3 2 0 0 2 0 2 ...
 $ ps_car_10_cat : int  1 1 1 1 1 1 1 1 1 1 ...
 $ ps_car_11_cat : int  12 19 60 104 82 104 99 30 68 104 ...
```

```
$ ps_car_11    : int  2 3 1 1 3 2 2 3 3 2 ...
$ ps_car_12    : num  0.4 0.316 0.316 0.374 0.316 ...
$ ps_car_13    : num  0.884 0.619 0.642 0.543 0.566 ...
$ ps_car_14    : num  0.371 0.389 0.347 0.295 0.365 ...
$ ps_car_15    : num  3.61 2.45 3.32 2 2 ...
$ ps_calc_01   : num  0.6 0.3 0.5 0.6 0.4 0.7 0.2 0.1 0.9 0.7 ...
$ ps_calc_02   : num  0.5 0.1 0.7 0.9 0.6 0.8 0.6 0.5 0.8 0.8 ...
$ ps_calc_03   : num  0.2 0.3 0.1 0.1 0 0.4 0.5 0.1 0.6 0.8 ...
$ ps_calc_04   : int  3 2 2 2 2 3 2 1 3 2 ...
$ ps_calc_05   : int  1 1 2 4 2 1 2 2 1 2 ...
$ ps_calc_06   : int  10 9 9 7 6 8 8 7 7 8 ...
$ ps_calc_07   : int  1 5 1 1 3 2 1 1 3 2 ...
$ ps_calc_08   : int  10 8 8 8 10 11 8 6 9 9 ...
$ ps_calc_09   : int  1 1 2 4 2 3 3 1 4 1 ...
$ ps_calc_10   : int  5 7 7 2 12 8 10 13 11 11 ...
$ ps_calc_11   : int  9 3 4 2 3 4 3 7 4 3 ...
$ ps_calc_12   : int  1 1 2 2 1 2 0 1 2 5 ...
$ ps_calc_13   : int  5 1 7 4 1 0 0 3 1 0 ...
$ ps_calc_14   : int  8 9 7 9 3 9 10 6 5 6 ...
$ ps_calc_15_bin: int  0 0 0 0 0 0 0 1 0 0 ...
$ ps_calc_16_bin: int  1 1 1 0 0 1 1 0 1 1 ...
$ ps_calc_17_bin: int  1 1 1 0 0 0 0 1 0 0 ...
$ ps_calc_18_bin: int  0 0 0 0 1 1 0 0 0 0 ...
$ ps_calc_19_bin: int  0 1 1 0 1 1 1 1 0 1 ...
$ ps_calc_20_bin: int  1 0 0 0 0 1 0 0 1 0 ...
```

str(test)

Classes ‘tbl_df’, ‘tbl’ and 'data.frame':	892816 obs. of 58 variables:

```
$ id           : int  0 1 2 3 4 5 6 8 10 11 ...
$ ps_ind_01    : int  0 4 5 0 5 0 0 0 0 1 ...
$ ps_ind_02_cat : int  1 2 1 1 1 1 1 1 1 1 ...
$ ps_ind_03    : int  8 5 3 6 7 6 3 0 7 6 ...
$ ps_ind_04_cat : int  1 1 0 0 0 0 0 0 0 0 ...
$ ps_ind_05_cat : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_06_bin : int  0 0 0 1 0 1 0 1 0 0 ...
$ ps_ind_07_bin : int  1 0 0 0 0 0 1 0 1 0 ...
$ ps_ind_08_bin : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_09_bin : int  0 1 1 0 1 0 0 0 0 1 ...
$ ps_ind_10_bin : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_11_bin : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_12_bin : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_13_bin : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_14    : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_ind_15    : int  12 5 10 4 4 10 11 7 6 7 ...
$ ps_ind_16_bin : int  1 1 0 1 1 1 0 1 1 0 ...
$ ps_ind_17_bin : int  0 0 0 0 0 0 1 0 0 1 ...
$ ps_ind_18_bin : int  0 0 0 0 0 0 0 0 0 0 ...
$ ps_reg_01    : num  0.5 0.9 0.4 0.1 0.9 0.9 0.1 0.9 0.4 0.9 ...
$ ps_reg_02    : num  0.3 0.5 0 0.2 0.4 0.5 0.1 1.1 0 1 ...
$ ps_reg_03    : num  0.61 0.771 0.916 NA 0.818 ...
$ ps_car_01_cat : int  7 4 11 7 11 9 6 7 11 11 ...
$ ps_car_02_cat : int  1 1 1 1 1 1 1 1 0 0 ...
$ ps_car_03_cat : int  NA NA NA NA NA NA NA NA 1 NA ...
```

```
 $ ps_car_04_cat : int  0 0 0 0 0 0 0 0 1 0 ...
 $ ps_car_05_cat : int  NA 0 NA NA NA NA 0 NA 0 NA ...
 $ ps_car_06_cat : int  1 11 14 1 11 11 1 11 2 4 ...
 $ ps_car_07_cat : int  1 1 1 1 1 0 1 1 NA 1 ...
 $ ps_car_08_cat : int  1 1 1 1 1 0 1 1 0 1 ...
 $ ps_car_09_cat : int  2 0 2 2 2 2 0 2 0 2 ...
 $ ps_car_10_cat : int  1 1 1 1 1 1 1 1 1 1 ...
 $ ps_car_11_cat : int  65 103 29 40 101 11 10 103 104 104 ...
 $ ps_car_11     : int  1 1 3 2 3 2 2 3 2 2 ...
 $ ps_car_12     : num  0.316 0.316 0.4 0.374 0.374 ...
 $ ps_car_13     : num  0.67 0.606 0.896 0.652 0.813 ...
 $ ps_car_14     : num  0.352 0.358 0.398 0.381 0.385 ...
 $ ps_car_15     : num  3.46 2.83 3.32 2.45 3.32 ...
 $ ps_calc_01    : num  0.1 0.4 0.6 0.1 0.9 0.7 0.9 0.8 0.9 0 ...
 $ ps_calc_02    : num  0.8 0.5 0.6 0.5 0.6 0.9 0.8 0.9 0.3 0.9 ...
 $ ps_calc_03    : num  0.6 0.4 0.6 0.5 0.8 0.4 0.8 0.5 0 0.7 ...
 $ ps_calc_04    : int  1 3 2 2 3 2 1 2 2 2 ...
 $ ps_calc_05    : int  1 3 3 1 4 1 1 2 2 1 ...
 $ ps_calc_06    : int  6 8 7 7 7 9 7 8 9 7 ...
 $ ps_calc_07    : int  3 4 4 3 1 5 3 4 7 1 ...
 $ ps_calc_08    : int  6 10 6 12 10 9 9 11 9 9 ...
 $ ps_calc_09    : int  2 2 3 1 4 4 5 2 0 1 ...
 $ ps_calc_10    : int  9 7 12 13 12 12 6 8 10 11 ...
 $ ps_calc_11    : int  1 2 4 5 4 8 2 3 5 6 ...
 $ ps_calc_12    : int  1 0 0 1 0 1 0 1 1 1 ...
 $ ps_calc_13    : int  1 3 2 0 0 4 4 4 4 6 ...
 $ ps_calc_14    : int  12 10 4 5 4 9 6 9 6 10 ...
 $ ps_calc_15_bin: int  0 0 0 1 0 1 1 0 0 0 ...
 $ ps_calc_16_bin: int  1 0 0 0 1 0 1 1 0 1 ...
 $ ps_calc_17_bin: int  1 1 0 1 1 1 0 0 1 1 ...
 $ ps_calc_18_bin: int  0 1 0 0 0 0 0 0 0 0 ...
 $ ps_calc_19_bin: int  0 0 0 0 0 1 0 0 0 0 ...
 $ ps_calc_20_bin: int  1 1 0 0 1 0 0 0 0 0 ...
```

The structure of the datasets says the data types of the variables are either numerical or integer. As per the data description,features that belong to similar groupings are tagged as such in the feature names (e.g., ind, reg, car, calc) and also the binary and categorical variables are postfixed as _bin and _cat respectively. Remaining features which are not tagged are either continuous or ordinal.

So, we have to convert the data types according to the given data description. I have done this step by row binding TEST and TRAIN sets together as we can do the pre-processing to the whole data at once.

```
## [1] 1488028 60
```

I have added a ‘target’ column to the TEST set (to make TEST set have same number of variables) and a ‘data’ column to the TEST and TRAIN sets which can be used to identify the test and train observations. The combined dataset has **1488028 Observations and 60 Variables.**

I used the following code to change the data type of variables

combined_data <- combined_data %>%
mutate_at(vars(ends_with("cat")), funs(as.factor)) %>%
mutate_at(vars(ends_with("bin")), funs(as.logical)) %>%
mutate(target = as.factor(target))

Now, let's see the structure of the data.

'data.frame': 1488028 obs. of 60 variables:

```
 $ id         : int 7 9 13 16 17 19 20 22 26 28 ...
 $ data       : chr "train" "train" "train" "train" ...
 $ target     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
 $ ps_ind_01    : int 2 1 5 0 0 5 2 5 5 1 ...
 $ ps_ind_02_cat : Factor w/ 4 levels "1","2","3","4": 2 1 4 1 2 1 1 1 1 1 ...
 $ ps_ind_03    : int 5 7 9 2 0 4 3 4 3 2 ...
 $ ps_ind_04_cat : Factor w/ 2 levels "0","1": 2 1 2 1 2 1 2 1 2 1 ...
 $ ps_ind_05_cat : Factor w/ 7 levels "0","1","2","3",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ ps_ind_06_bin : logi FALSE FALSE FALSE TRUE TRUE FALSE ...
 $ ps_ind_07_bin : logi TRUE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_ind_08_bin : logi FALSE TRUE TRUE FALSE FALSE FALSE ...
 $ ps_ind_09_bin : logi FALSE FALSE FALSE FALSE FALSE TRUE ...
 $ ps_ind_10_bin : logi FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_ind_11_bin : logi FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_ind_12_bin : logi FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_ind_13_bin : logi FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_ind_14    : int 0 0 0 0 0 0 0 0 0 0 ...
 $ ps_ind_15    : int 11 3 12 8 9 6 8 13 6 4 ...
 $ ps_ind_16_bin : logi FALSE FALSE TRUE TRUE TRUE TRUE ...
 $ ps_ind_17_bin : logi TRUE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_ind_18_bin : logi FALSE TRUE FALSE FALSE FALSE FALSE ...
 $ ps_reg_01    : num 0.7 0.8 0 0.9 0.7 0.9 0.6 0.7 0.9 0.9 ...
 $ ps_reg_02    : num 0.2 0.4 0 0.2 0.6 1.8 0.1 0.4 0.7 1.4 ...
 $ ps_reg_03    : num 0.718 0.766 NA 0.581 0.841 ...
 $ ps_car_01_cat : Factor w/ 12 levels "0","1","2","3",..: 11 12 8 8 12 11 7 12 11 12 ...
 $ ps_car_02_cat : Factor w/ 2 levels "0","1": 2 2 2 2 2 1 2 2 2 1 ...
 $ ps_car_03_cat : Factor w/ 2 levels "0","1": NA NA NA 1 NA NA NA 1 NA 1 ...
 $ ps_car_04_cat : Factor w/ 10 levels "0","1","2","3",..: 1 1 1 1 1 1 1 1 1 2 ...
 $ ps_car_05_cat : Factor w/ 2 levels "0","1": 2 NA NA 2 NA 1 2 1 2 1 ...
 $ ps_car_06_cat : Factor w/ 18 levels "0","1","2","3",..: 5 12 15 12 15 15 12 12 15 15 ...
 $ ps_car_07_cat : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
 $ ps_car_08_cat : Factor w/ 2 levels "0","1": 1 2 2 2 2 2 2 2 2 2 ...
 $ ps_car_09_cat : Factor w/ 5 levels "0","1","2","3",..: 1 3 3 4 3 1 1 3 1 3 ...
 $ ps_car_10_cat : Factor w/ 3 levels "0","1","2": 2 2 2 2 2 2 2 2 2 2 ...
 $ ps_car_11_cat : Factor w/ 104 levels "1","2","3","4",..: 12 19 60 104 82 104 99 30 68 104 ...
 $ ps_car_11    : int 2 3 1 1 3 2 2 3 3 2 ...
 $ ps_car_12    : num 0.4 0.316 0.316 0.374 0.316 ...
 $ ps_car_13    : num 0.884 0.619 0.642 0.543 0.566 ...
```

```
 $ ps_car_14    : num  0.371 0.389 0.347 0.295 0.365 ...
 $ ps_car_15    : num  3.61 2.45 3.32 2 2 ...
 $ ps_calc_01   : num  0.6 0.3 0.5 0.6 0.4 0.7 0.2 0.1 0.9 0.7 ...
 $ ps_calc_02   : num  0.5 0.1 0.7 0.9 0.6 0.8 0.6 0.5 0.8 0.8 ...
 $ ps_calc_03   : num  0.2 0.3 0.1 0.1 0 0.4 0.5 0.1 0.6 0.8 ...
 $ ps_calc_04   : int  3 2 2 2 2 3 2 1 3 2 ...
 $ ps_calc_05   : int  1 1 2 4 2 1 2 2 1 2 ...
 $ ps_calc_06   : int  10 9 9 7 6 8 8 7 7 8 ...
 $ ps_calc_07   : int  1 5 1 1 3 2 1 1 3 2 ...
 $ ps_calc_08   : int  10 8 8 8 10 11 8 6 9 9 ...
 $ ps_calc_09   : int  1 1 2 4 2 3 3 1 4 1 ...
 $ ps_calc_10   : int  5 7 7 2 12 8 10 13 11 11 ...
 $ ps_calc_11   : int  9 3 4 2 3 4 3 7 4 3 ...
 $ ps_calc_12   : int  1 1 2 2 1 2 0 1 2 5 ...
 $ ps_calc_13   : int  5 1 7 4 1 0 0 3 1 0 ...
 $ ps_calc_14   : int  8 9 7 9 3 9 10 6 5 6 ...
 $ ps_calc_15_bin: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ ps_calc_16_bin: logi  TRUE TRUE TRUE FALSE FALSE TRUE ...
 $ ps_calc_17_bin: logi  TRUE TRUE TRUE FALSE FALSE FALSE ...
 $ ps_calc_18_bin: logi  FALSE FALSE FALSE FALSE TRUE TRUE ...
 $ ps_calc_19_bin: logi  FALSE TRUE TRUE FALSE TRUE TRUE ...
 $ ps_calc_20_bin: logi  TRUE FALSE FALSE FALSE FALSE TRUE ...
```

Let's explore the missing values of the data. We have already seen there are 846458 and 1270295 missing values in TRAIN and TEST data respectively. Below are the column wise missing values in combined data (both TEST and TRAIN)

```
           colSums(is.na(combined_data))
id                               0
data                             0
target                           0
ps_ind_01                        0
ps_ind_02_cat                  523
ps_ind_03                        0
ps_ind_04_cat                  228
ps_ind_05_cat                14519
ps_ind_06_bin                    0
ps_ind_07_bin                    0
ps_ind_08_bin                    0
ps_ind_09_bin                    0
ps_ind_10_bin                    0
ps_ind_11_bin                    0
ps_ind_12_bin                    0
ps_ind_13_bin                    0
ps_ind_14                        0
ps_ind_15                        0
ps_ind_16_bin                    0
ps_ind_17_bin                    0
ps_ind_18_bin                    0
ps_reg_01                        0
ps_reg_02                        0
ps_reg_03                   269456
ps_car_01_cat                  267
ps_car_02_cat                   10
ps_car_03_cat              1028142
```

ps_car_04_cat	0
ps_car_05_cat	666910
ps_car_06_cat	0
ps_car_07_cat	28820
ps_car_08_cat	0
ps_car_09_cat	1446
ps_car_10_cat	0
ps_car_11_cat	0
ps_car_11	6
ps_car_12	1
ps_car_13	0
ps_car_14	106425
ps_car_15	0
ps_calc_01	0
ps_calc_02	0
ps_calc_03	0
ps_calc_04	0
ps_calc_05	0
ps_calc_06	0
ps_calc_07	0
ps_calc_08	0
ps_calc_09	0
ps_calc_10	0
ps_calc_11	0
ps_calc_12	0
ps_calc_13	0
ps_calc_14	0
ps_calc_15_bin	0
ps_calc_16_bin	0
ps_calc_17_bin	0
ps_calc_18_bin	0
ps_calc_19_bin	0
ps_calc_20_bin	0

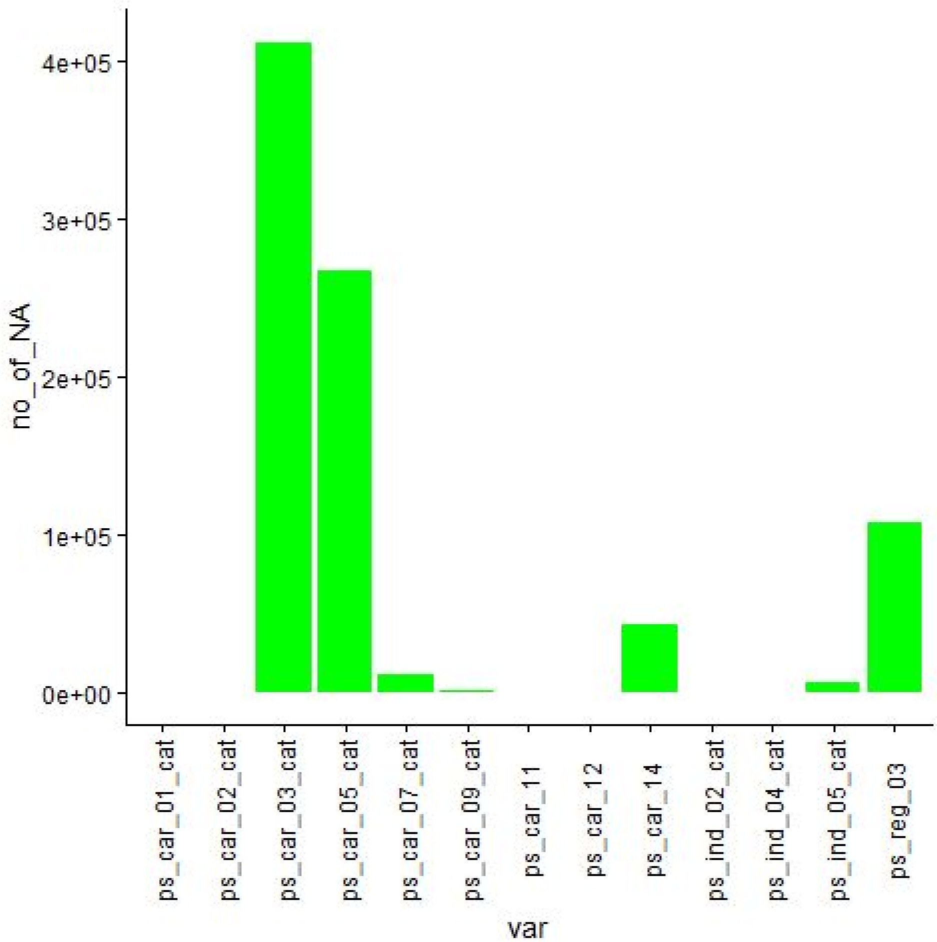
Columns with missing values:

"ps_ind_02_cat" "ps_ind_04_cat" "ps_ind_05_cat" "ps_reg_03" "ps_car_01_cat" "ps_car_02_cat" "ps_car_03_cat" "ps_car_05_cat" "ps_car_07_cat" "ps_car_09_cat" "ps_car_11" "ps_car_12" "ps_car_14"

There are a lot of concentration of missing values in few columns. Let's drop them with a threshold percentage. I am dropping variable with >=5% of missing values in them.

[1] "Columns with >=5% of missing values are:"
[1] "ps_reg_03" "ps_car_03_cat" "ps_car_05_cat" "ps_car_14"
[1] "Dimensions after dropping the variables with >=5% of missing values:"
[1] 1488028 56

There are missing values to be imputed in these remaining variables. I am imputing NAs with 'mode' in categorical/factor variable and with 'mean' in numerical variables.

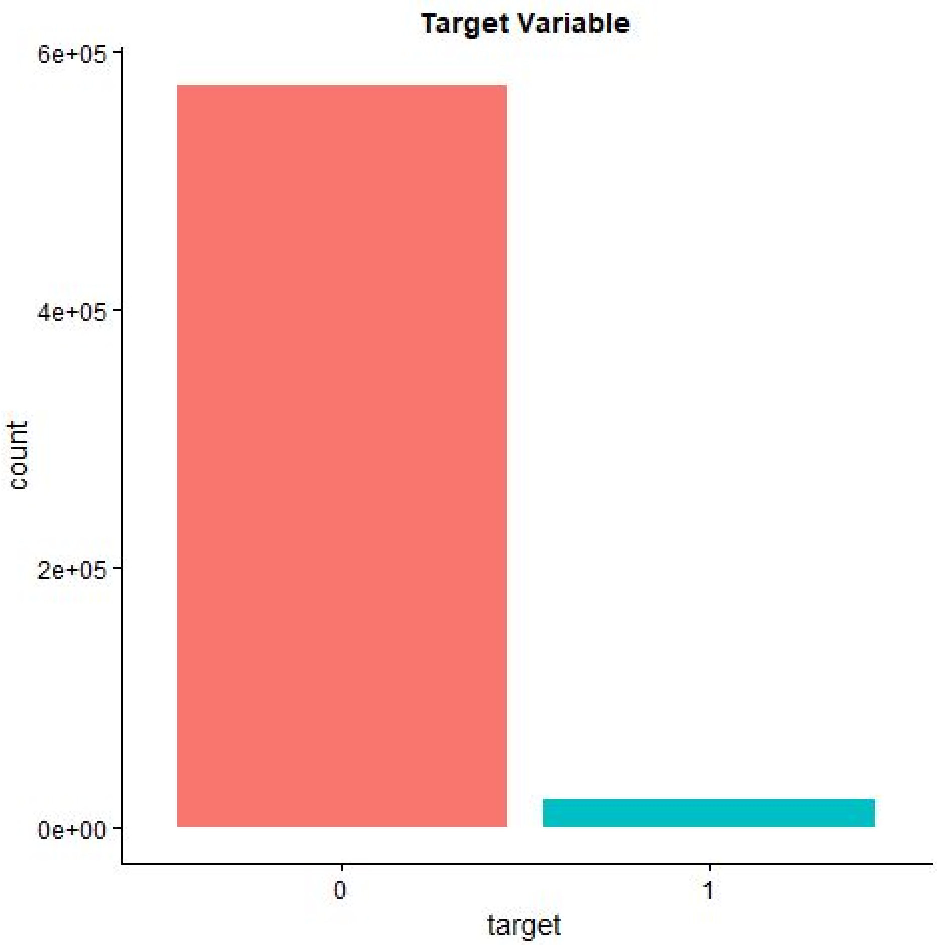


We have converted the data types as required and imputed the missing values. Now, split the combined data back to TRAIN and TEST sets.

```
## [1] "Dimensions of TRAIN:"
## [1] 595212 55
## [1] "Dimensions of TEST"
## [1] 892816 54
```

Feature Engineering

Let's analyze the target variable in TRAIN.




```
##
## 0       1
## 573518 21694

##
## 0              1
## 0.96355248 0.03644752
```

We can see that the target variable is imbalanced. Class 0 have 0.963% observations & 1 have 0.037% observations. Let's balance the data set using over/under sampling.

```
## [1] "Dimensions of data after balancing the target variable"
## [1] 90000 55
## [1] "No of missing values:"
## [1] 0
## [1] "Balance of the target variable:"
##
## 0 1
```

44959 45041
##
0 1
0.4995444 0.5004556

Now the target variable is balanced in sampled out data frame with 90000 observations. We can take whole data set but it requires more computational power. And more over using over and under sampling technique, if we draw a large data set, the algorithm may drop contributing observations. We can use 'ROSE' function but it works only on factor and numerical data not on ordinal data.

Machine Learning (Model Building)

I am building a Random Forest model to predict the target variable. To check the efficiency of the model, I am diving the TRAIN set to train and test sets with a ratio of 7:3(train:test).
[1] 63000 55
[1] 27000 55
The 'train' set dimensions are 63000 X 55 and 'test' set dimensions are 27000 X 55.

1. **Random Forest**

Random forest can take only maximum 53 levels in a variable. "ps_car_11_cat" has 104 levels in total. So, it has to be removed to build a random forest model.
[1] 63000 54
[1] 27000 54

Random Forest is giving the below results.

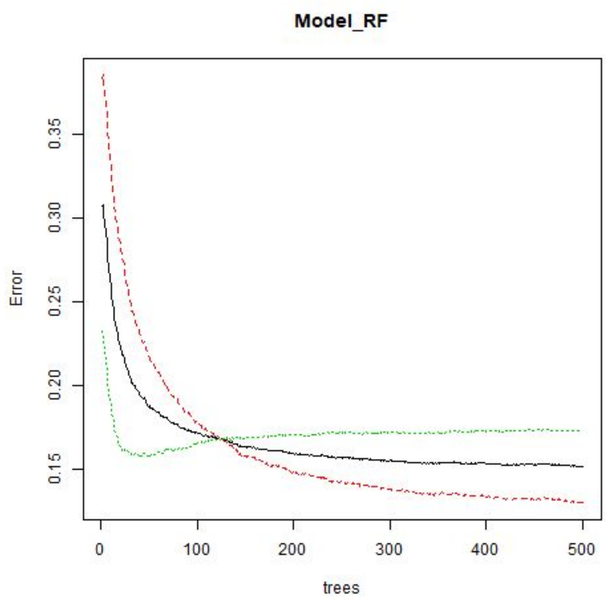
Confusion Matrix and Statistics

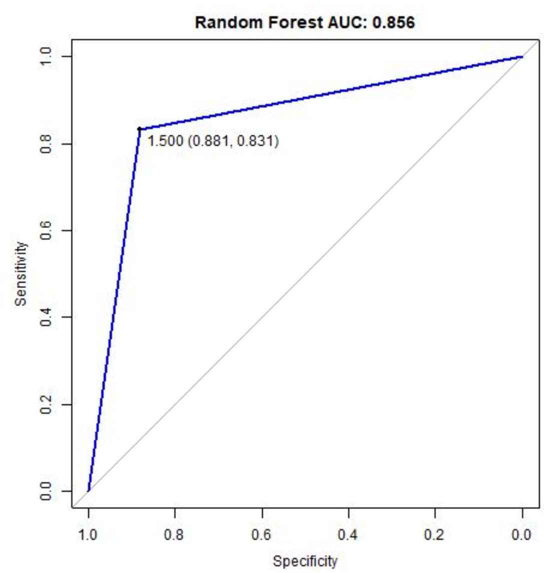
```
        Reference
Prediction     0     1
         0 11935  2269
         1  1610 11186

              Accuracy : 0.8563
                95% CI : (0.8521, 0.8605)
   No Information Rate : 0.5017
   P-Value [Acc > NIR] : < 2.2e-16
```

Kappa : 0.7126
Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.8811
Specificity : 0.8314
Pos Pred Value : 0.8403
Neg Pred Value : 0.8742
Prevalence : 0.5017
Detection Rate : 0.4420
Detection Prevalence : 0.5261
Balanced Accuracy : 0.8563

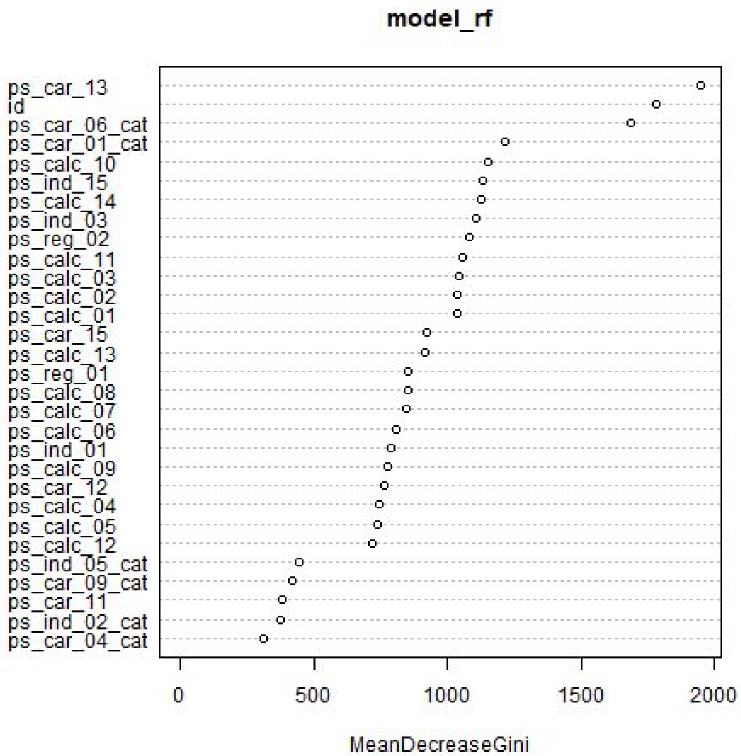
'Positive' Class : 0





We got an accuracy of 85.63 % from the above random forest model.

Important variables plot from this random forest model



Using the variable importance, I included only the important variable and build second random forest model, below are the improvements.

Results of tuned Random Forests are below,

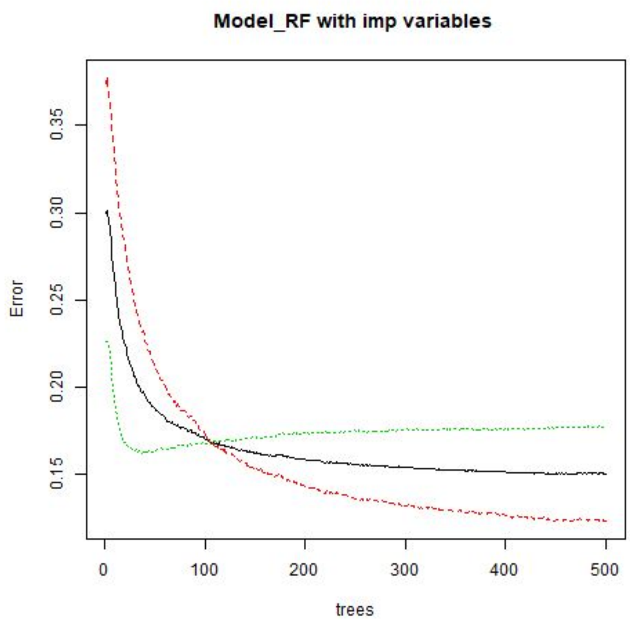
Confusion Matrix and Statistics

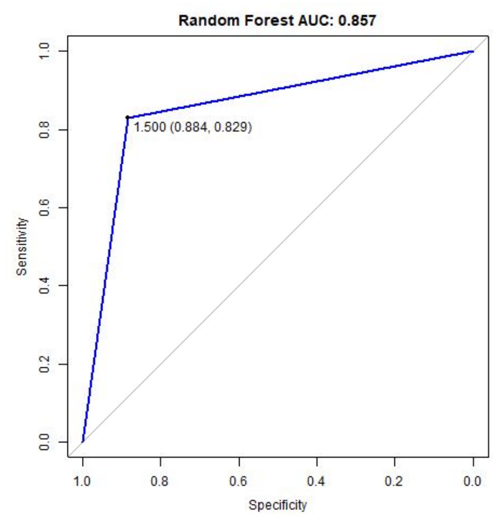
```
          Reference
Prediction     0     1
         0 11976  2295
         1  1569 11160

               Accuracy : 0.8569
                 95% CI : (0.8527, 0.861)
    No Information Rate : 0.5017
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7137
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8842
            Specificity : 0.8294
         Pos Pred Value : 0.8392
```

Neg Pred Value : 0.8767
Prevalence : 0.5017
Detection Rate : 0.4436
Detection Prevalence : 0.5286
Balanced Accuracy : 0.8568

'Positive' Class : 0

We see the fluctuation of 0.2-0.3 % in the accuracy from selecting the important variables.





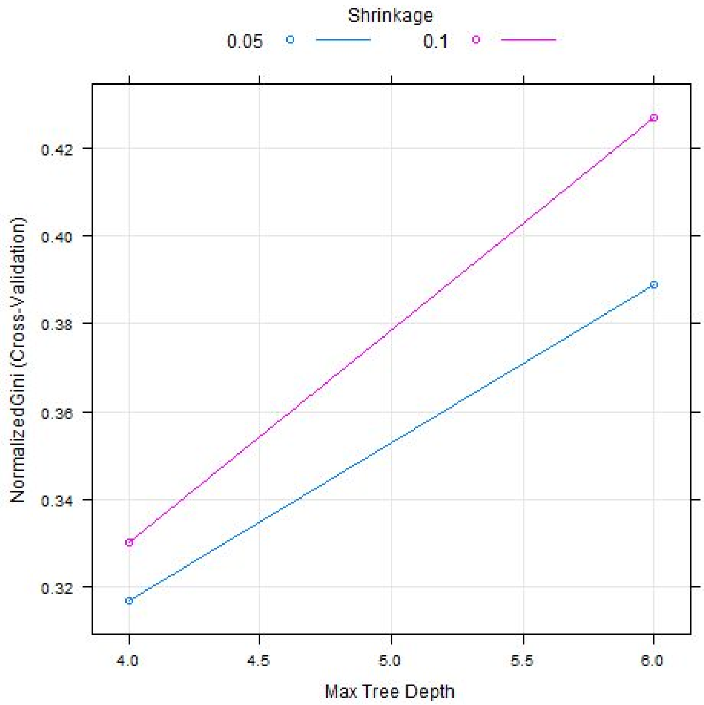
. Random Forests gave us 85% percent of accuracy.

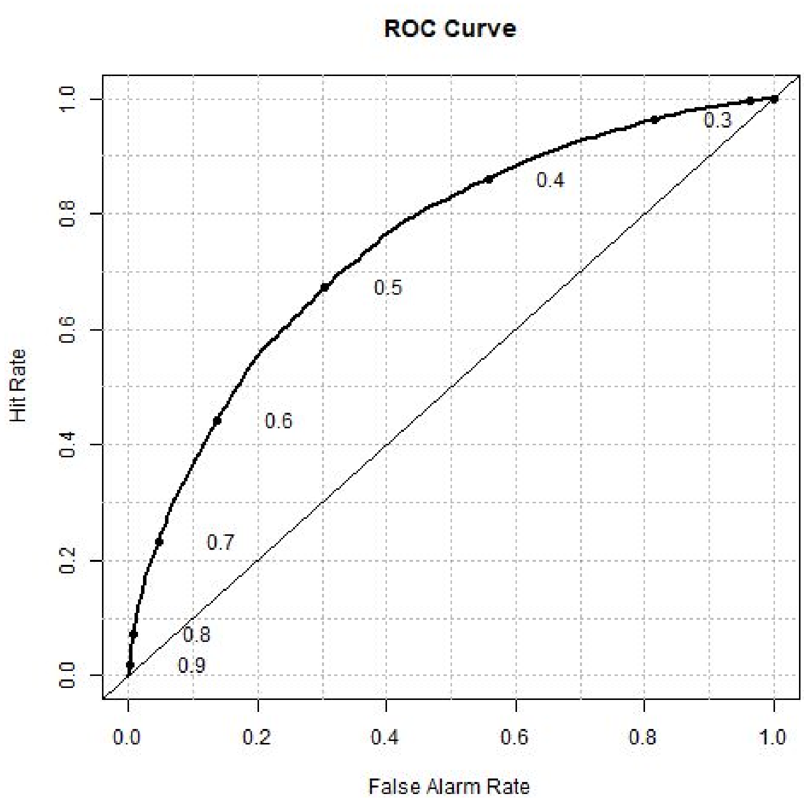
The Gini Coefficient ranges from approximately 0 for random guessing, to approximately 0.5 for a perfect score. The theoretical maximum for the discrete calculation is (1 - frac_pos) / 2.

2. **Xgboost**

Even I experimented with **xgboost** model which gave an accuracy of 68%.

```
          Reference
Prediction   No  Yes
       No  9270 4417
       Yes 4112 9201

               Accuracy : 0.6841
                 95% CI : (0.6785, 0.6897)
    No Information Rate : 0.5044
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3683
 Mcnemar's Test P-Value : 0.0009957

            Sensitivity : 0.6927
            Specificity : 0.6756
         Pos Pred Value : 0.6773
         Neg Pred Value : 0.6911
             Prevalence : 0.4956
         Detection Rate : 0.3433
   Detection Prevalence : 0.5069
      Balanced Accuracy : 0.6842

       'Positive' Class : No
```


Shrinkage
0.05
0.1
0.42
0.40
0.38
0.36
0.34
0.32
NormalizedGini (Cross-Validation)
4.0
4.5
5.0
5.5
6.0
Max Tree Depth


ROC Curve
Hit Rate
False Alarm Rate
0.0
0.2
0.4
0.6
0.8
1.0
0.3
0.4
0.5
0.6
0.7
0.8
0.9

xgbTree variable importance

Only 20 most important variables shown (out of 204)

	Overall
ps_car_13	*100.00*
id	*89.48*
ps_ind_15	*39.60*
ps_ind_03	*39.02*
ps_reg_02	*38.46*
ps_calc_10	*29.06*
ps_calc_03	*28.18*
ps_calc_14	*28.12*
ps_calc_01	*24.90*
ps_reg_01	*24.65*
ps_ind_01	*24.34*
ps_calc_02	*24.30*
ps_calc_11	*22.99*
ps_calc_13	*21.00*
ps_car_12	*19.71*
ps_calc_07	*18.88*
ps_car_15	*18.76*
ps_calc_08	*16.71*
ps_calc_09	*16.58*
ps_calc_05	*15.55*

As experiments, I have build model with ***Naive Bayes***, ***Gradient Boosting algorithms***. Naive Bayes gave an accuracy of 58% where as the GBM gave 59% of accuracy.

Thus, I'm freezing the random forest model as it gives 85% accuracy.

The head of the final submission file:

	id	target
1:	*0*	*0.312*
2:	*1*	*0.326*
3:	*2*	*0.378*

4: 3 0.222
5: 4 0.398

The results for the test data given is submitted as submission.csv file.

THANK YOU…