FIT5149 - Advanced Data Analysis - Assessment 1

Choosing and Explaining Likely Caravan Insurance Customers

Name: Girish D Bhatta

StudentID: 29270863

Email: gbha0005@student.monash.edu

Overview

1. Objective

*The objective of this assignment is to come up with a **Proabilistic Model** which predicts the likelihood of a customer opting for a Caravan Insurance Policy. Furthermore, the larger motive of this problem is also to identify what are the main contributing factors that lead to a customer opting for a Caravan Policy. This is particularly essential considering this could serve as a prominent reference point to the Company and especially to its Marketing Division and get a clearer picture as to what section or attributes has to be targeted in the cross section of the customer to achieve more revenue and eventually increase the customer*

2. About the Dataset

*The dataset provides information about customers about Caravan insurance policy. The dataset provided has 86 features in total. features **1-43** provide socio-demographic inforamtion about the customer and features **44-86** provide information about the product ownership . The 86th feature tells if the customer bought the Caravan policy or not. 0 represents **No** and 1 represents **Yes**. The dataset is divided into 2 parts : Train and Test dataset. However, the task is to predict the probablity of a customer buying a Caravan policy of the Test Dataset by learning a model from Training Dataset. Furthermore, we can evaluate the accuracy of the predictions by using a confusion matrix provided as the evaluation test set. Additionally, we also have to provide details about the features that contribute the most in the prediction. This is important for the marketing team to make smart decisions in their investment in order to achieve good ROI(Returns on Investment)*

3. Approach/Methodology

Advanced Data Analysis is a branch of Data Science which boasts of most diverse and unconventional methods and approaches to Solve a Machine learning problem. The same problem could be potentially solved in umpteen number of ways and the methodology changes from person to person. However, in this particular problem, we will come with a more streamlined approach and employ an applied way of analysing data, which might as well involve interpretation of the feature, considering the description of each feature and understanding the feature's behabviour in terms of the response variable. The outline of the methodology would be as follows:

- ***Getting rid of Zero/nearZeroVariance variables**: This is particularly important in data analysis as it is statistically insignificat to include features that exhibit little to NO variance at all. The response variable cannot or rather the Probabilistic model cannot me made interpretable or separable in terms of the features if the predictors used show no variance. Therefore, removing these variables would reduce the subset of features to be considered considerably small and hence will make the EDA process more feasible*

- ***Removing features with high perentage of missing values** : In this step we would examine features for missing values and ascertain the percatage of missing values(if any). This is essential as features with high percentage of missing values do not contribute much to the probabilistic model and doesnt not provide enough information about the behaviour of the feature with respect to the response variable. Therefore, this is one of the first steps towards reducing dimensions in a dataset*

- ***Performing detailed Exploratory Data Analysis for all the Features** : In this stage we perform detailed Exploratory Data Analysis for each of the feature in the dataset. As part of Exploration, we will include several plots to make the exploratory process more insightful. It will include distribution of the feature, distribution of the feature with respect to the response variable which would indicate the degree of correlation between the feature and response variable. Box-plots and Scatter plots will also be included as part of Exploration. The EDA would serve as a prominent checkpoint for us to form the hypothesis of the likelihood correlation of the feature with the target.*

- ***Getting rid of pair-wise correlation among the predictors** : For a data model that takes several features as predictors as the confoudnding parts of the problem, it is not desirable to have pair-wise correlation among the predictors. The simple reason being, the more number of predictors the model learns on, more are the chances of the model over fitting. With increase in predictors the model will show better accuracy(R-squared value). However, it will underperform(RMSE will increase) on unseen data(test data). Furthermore, the aim is to make our model as general and as interpretable as possible and generality is compromised if more features are added to the training. Also, it is always advisable to add a single feature to the model, when any two given features are correlated. This also reduces the redundancy of the model. This particulary enhances the model interpretability and generality*

- ***Identifying features highly correlated to the response variable** : For a predictive model to be able to have a good true positive rate should have a good collection of features that are highly correlated to the response variable. These features would then be used in various models to ascertain the importance or the significance of this feature towards the response variable. However, there are chances that there are features which are correlated among themseleves and are highly correlated with the response variable as well. We must be cautious while getting rid of pari-wise correlation features as an interaction among the these features can lead to a better predictor for the response variable. Therefore, there has to be a double checking mechanism between this and the previous step.*

- ***Wrapper methods** : In most of the data science projects, the larger objective is to reduce the dimensionality of the dataset to a handful subset of features which describe the response more prominently. Therefore, Curse of Dimensionality is a challenging problem to face and we use*

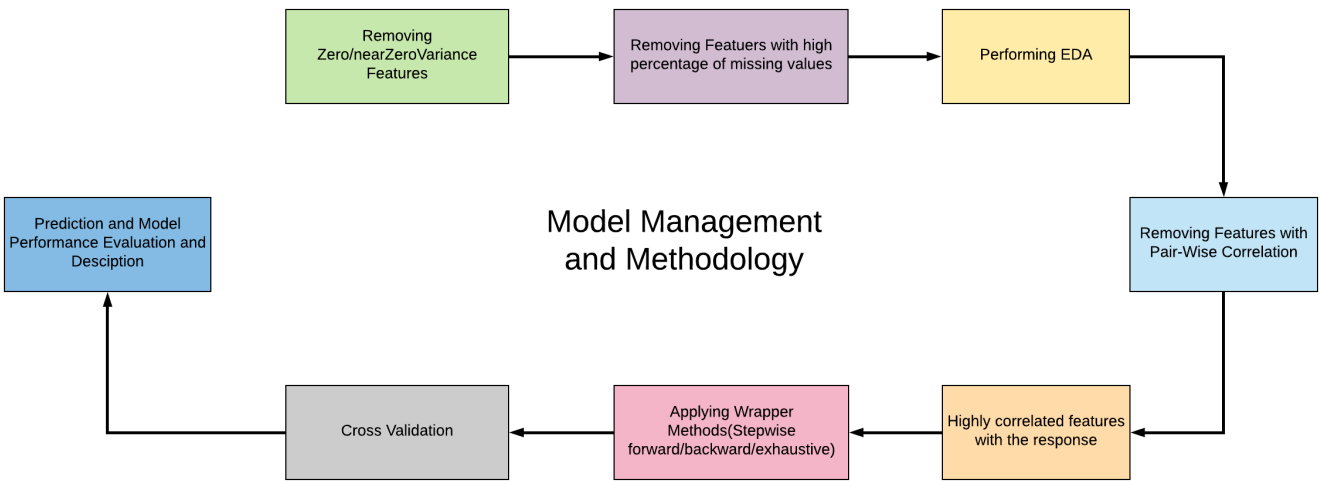
wrapper methods at our disposal to reduce the dimesnsions of the dataset. We will use a wrapper method of forward, backward and exhuative method which will select the most prominent of features over several iterations. These features will be then used in cross validation methods to understand the significance of the features used. Therefore, arrving at a smaller subset of varibales is the goal of this particular step.

- ***Cross Validation** : After we have selected a bunch of features which potentially provide promising predictions, we have to make sure that the set of features selected perform robustly with the model selected. One way fo ensuring is to perform cross validation wherein, the dataset will be divided into k-folds where k-1 folds will be used for training and the remaining one fold for testing. This will essentailly ensure the model has circumvented several folds of data thereby making it more robust and susceptible and lesser chance of producing unreliable results when fed with unseen data.*

- ***Prediction of Carvan Policies for Customers** : After we have found out the model performs robustly and behaves well with unseen data, we use the subset of features to predict the probability of the customer likely to buy a caravan policy. We can later on evaluate our results with the evaluation test data. The same procedure has to be repeated for other models as well.*

- ***Evaluation of Model Performance** : Lastly, we evaluate the model performace in terms of accuracy, specificity and other model evaluation metrics. Furthermore, examining the features for true positive predictions and provide a description about the features which have contributed towards the prediction of these results. This would also serve as a reference point for the marketing team to look for customer with certain attributes in order to maximize the revenue of the company.*

The methods above discussed, have been abstracted in a flow chart below.



Implementation

1. Importing Libraries and Reading the Dataset

In [263]:

```
library(corrplot)
library(ggplot2)
library(lattice)
library(caret)
library(leaps)
library(pROC)
library(klaR)
library(e1071)
library(plotrix)
```

```
Attaching package: 'plotrix'

The following object is masked from 'package:psych':

    rescale
```

In [2]:

```
# reading the column names from a csv file
columns <- read.csv("column_names.csv",header = FALSE)

# reading the training dataset
train_data <- read.delim("ticdata2000.txt",sep = "\t",header = FALSE)

# reading the evalutation dataset
eval_data <- read.delim2("tictgts2000.txt",header = FALSE)

# reading the test dataset
test_data <- read.delim2("ticeval2000.txt",header = FALSE)
```

1.1 Data preparation

In this section we are going to check if the data read is already in the right format. If not, we will perform some Data wrangling to get it into correct format. let us first inspect how the data looks.

In [3]:

```
head(train_data)
```

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	⋯	V77	V78	V79	V80	V81	V82	V83	V84	V8
33	1	3	2	8	0	5	1	3	7	⋯	0	0	0	1	0	0	0	0	0
37	1	2	2	8	1	4	1	4	6	⋯	0	0	0	1	0	0	0	0	0
37	1	2	2	8	0	4	2	4	3	⋯	0	0	0	1	0	0	0	0	0
9	1	3	3	3	2	3	2	4	5	⋯	0	0	0	1	0	0	0	0	0
40	1	4	2	10	1	4	1	4	7	⋯	0	0	0	1	0	0	0	0	0
23	1	2	1	5	0	5	0	5	0	⋯	0	0	0	0	0	0	0	0	0

In [4]:

```
dim(train_data)
```

```
5822  86
```

We can see that there are 86 columns in the dataframe. The training dataset has 5822 rows and 86 columns in total. We can see that the columns are named as numbers and not what attrubute they actually represent. We can use regular expressions and extract the Column names from the columns dataframe and apply those column names to this dataframe.

In [5]:

```
extract = c()
pattern <- "^[[:space:]]?[[:upper:]]+"
for (variable in columns$V1) {
  extract <- c(extract, regmatches(variable, regexpr(pattern, variable)))
}
columns$V1 <- extract
columns$V1 <- make.names(columns$V1, unique=TRUE)
```

In [6]:

```
#assigning the column names to train data
names(train_data) <- columns$V1
names(test_data) <- columns$V1[1:85]
```

In [7]:

```
head(train_data)
```

X.MOSTYPE	X.MAANTHUI	X.MGEMOMV	X.MGEMLEEF	X.MOSHOOFD	X.MGODRK
33	1	3	2	8	0
37	1	2	2	8	1
37	1	2	2	8	0
9	1	3	3	3	2
40	1	4	2	10	1
23	1	2	1	5	0

We can observe that the column names have been changed to more descriptive column names.

2. Preliminary Analysis

Let us have a look at the summary of the train dataset

In [8]:

```
summary(train_data)
```

```
   X.MOSTYPE       X.MAANTHUI       X.MGEMOMV       X.MGEMLEEF
 Min.   : 1.00   Min.   : 1.000   Min.   :1.000   Min.   :1.000
 1st Qu.:10.00   1st Qu.: 1.000   1st Qu.:2.000   1st Qu.:2.000
 Median :30.00   Median : 1.000   Median :3.000   Median :3.000
 Mean   :24.25   Mean   : 1.111   Mean   :2.679   Mean   :2.991
 3rd Qu.:35.00   3rd Qu.: 1.000   3rd Qu.:3.000   3rd Qu.:3.000
 Max.   :41.00   Max.   :10.000   Max.   :5.000   Max.   :6.000
   X.MOSHOOFD        X.MGODRK         X.MGODPR        X.MGODOV
 Min.   : 1.000   Min.   :0.0000   Min.   :0.000   Min.   :0.00
 1st Qu.: 3.000   1st Qu.:0.0000   1st Qu.:4.000   1st Qu.:0.00
 Median : 7.000   Median :0.0000   Median :5.000   Median :1.00
 Mean   : 5.774   Mean   :0.6965   Mean   :4.627   Mean   :1.07
 3rd Qu.: 8.000   3rd Qu.:1.0000   3rd Qu.:6.000   3rd Qu.:2.00
 Max.   :10.000   Max.   :9.0000   Max.   :9.000   Max.   :5.00
    X.MGODGE        X.MRELGE         X.MRELSA         X.MRELOV
 Min.   :0.000   Min.   :0.000   Min.   :0.0000   Min.   :0.00
 1st Qu.:2.000   1st Qu.:5.000   1st Qu.:0.0000   1st Qu.:1.00
 Median :3.000   Median :6.000   Median :1.0000   Median :2.00
 Mean   :3.259   Mean   :6.183   Mean   :0.8835   Mean   :2.29
 3rd Qu.:4.000   3rd Qu.:7.000   3rd Qu.:1.0000   3rd Qu.:3.00
 Max.   :9.000   Max.   :9.000   Max.   :7.0000   Max.   :9.00
   X.MFALLEEN      X.MFGEKIND     X.MFWEKIND    X.MOPLHOOG       X.MO
PLMIDD
 Min.   :0.000   Min.   :0.00   Min.   :0.0   Min.   :0.000   Min.
:0.000
 1st Qu.:0.000   1st Qu.:2.00   1st Qu.:3.0   1st Qu.:0.000   1st Q
u.:2.000
 Median :2.000   Median :3.00   Median :4.0   Median :1.000   Median
:3.000
 Mean   :1.888   Mean   :3.23   Mean   :4.3   Mean   :1.461   Mean
:3.351
 3rd Qu.:3.000   3rd Qu.:4.00   3rd Qu.:6.0   3rd Qu.:2.000   3rd Q
u.:4.000
 Max.   :9.000   Max.   :9.00   Max.   :9.0   Max.   :9.000   Max.
:9.000
   X.MOPLLAAG      X.MBERHOOG      X.MBERZELF      X.MBERBOER
 Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.0000
 1st Qu.:3.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.0000
 Median :5.000   Median :2.000   Median :0.000   Median :0.0000
 Mean   :4.572   Mean   :1.895   Mean   :0.398   Mean   :0.5223
 3rd Qu.:6.000   3rd Qu.:3.000   3rd Qu.:1.000   3rd Qu.:1.0000
 Max.   :9.000   Max.   :9.000   Max.   :5.000   Max.   :9.0000
   X.MBERMIDD      X.MBERARBG     X.MBERARBO        X.MSKA
X.MSKB
 Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.
:0.000
 1st Qu.:2.000   1st Qu.:1.00   1st Qu.:1.000   1st Qu.:0.000   1st
Qu.:1.000
 Median :3.000   Median :2.00   Median :2.000   Median :1.000   Medi
an :2.000
 Mean   :2.899   Mean   :2.22   Mean   :2.306   Mean   :1.621   Mean
:1.607
 3rd Qu.:4.000   3rd Qu.:3.00   3rd Qu.:3.000   3rd Qu.:2.000   3rd
Qu.:2.000
 Max.   :9.000   Max.   :9.00   Max.   :9.000   Max.   :9.000   Max.
:9.000
    X.MSKB.1         X.MSKC          X.MSKD         X.MHHUUR
 Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
 1st Qu.:1.000   1st Qu.:2.000   1st Qu.:0.000   1st Qu.:2.000
 Median :2.000   Median :4.000   Median :1.000   Median :4.000
 Mean   :2.203   Mean   :3.759   Mean   :1.067   Mean   :4.237
```

```
 3rd Qu.:3.000   3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:7.000
 Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
    X.MHKOOP         X.MAUT        X.MAUT.1        X.MAUT.2       X.
MZFONDS
 Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.
:0.000
 1st Qu.:2.000   1st Qu.:5.00   1st Qu.:0.000   1st Qu.:1.000   1st
Qu.:5.000
 Median :5.000   Median :6.00   Median :1.000   Median :2.000   Medi
an :7.000
 Mean   :4.772   Mean   :6.04   Mean   :1.316   Mean   :1.959   Mean
:6.277
 3rd Qu.:7.000   3rd Qu.:7.00   3rd Qu.:2.000   3rd Qu.:3.000   3rd
Qu.:8.000
 Max.   :9.000   Max.   :9.00   Max.   :7.000   Max.   :9.000   Max.
:9.000
    X.MZPART        X.MINKM          X.MINK         X.MINK.1
 Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
 1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:1.000
 Median :2.000   Median :2.000   Median :4.000   Median :3.000
 Mean   :2.729   Mean   :2.574   Mean   :3.536   Mean   :2.731
 3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:5.000   3rd Qu.:4.000
 Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
    X.MINK.2         X.MINK.3        X.MINKGEM       X.MKOOPKLA
 Min.   :0.0000   Min.   :0.0000   Min.   :0.000   Min.   :1.000
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:3.000
 Median :0.0000   Median :0.0000   Median :4.000   Median :4.000
 Mean   :0.7961   Mean   :0.2027   Mean   :3.784   Mean   :4.236
 3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:4.000   3rd Qu.:6.000
 Max.   :9.0000   Max.   :9.0000   Max.   :9.000   Max.   :8.000
   X.PWAPART        X.PWABEDR         X.PWALAND         X.PPERSAUT
 Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00
 Median :0.0000   Median :0.00000   Median :0.00000   Median :5.00
 Mean   :0.7712   Mean   :0.04002   Mean   :0.07162   Mean   :2.97
 3rd Qu.:2.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:6.00
 Max.   :3.0000   Max.   :6.00000   Max.   :4.00000   Max.   :8.00
   X.PBESAUT        X.PMOTSCO        X.PVRAAUT          X.PAANHANG
 Min.   :0.00000   Min.   :0.0000   Min.   :0.000000   Min.   :0.000
00
 1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.000000   1st Qu.:0.000
00
 Median :0.00000   Median :0.0000   Median :0.000000   Median :0.000
00
 Mean   :0.04827   Mean   :0.1754   Mean   :0.009447   Mean   :0.020
96
 3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.000000   3rd Qu.:0.000
00
 Max.   :7.00000   Max.   :7.0000   Max.   :9.000000   Max.   :5.000
00
   X.PTRACTOR        X.PWERKT          X.PBROM         X.PLEVEN
 Min.   :0.00000   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
 1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.0000
 Median :0.00000   Median :0.00000   Median :0.000   Median :0.0000
 Mean   :0.09258   Mean   :0.01305   Mean   :0.215   Mean   :0.1948
 3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.000   3rd Qu.:0.0000
 Max.   :6.00000   Max.   :6.00000   Max.   :6.000   Max.   :9.0000
   X.PPERSONG        X.PGEZONG         X.PWAOREG         X.PBRAND
 Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.000
 1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.000
 Median :0.00000   Median :0.00000   Median :0.00000   Median :2.000
```

```
 Mean   :0.01374   Mean   :0.01529   Mean   :0.02353   Mean   :1.828
 3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:4.000
 Max.   :6.00000   Max.   :3.00000   Max.   :7.00000   Max.   :8.000
   X.PZEILPL          X.PPLEZIER         X.PFIETS          X.PINBOED
 Min.   :0.0000000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0
0000
 1st Qu.:0.0000000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0
0000
 Median :0.0000000   Median :0.00000   Median :0.00000   Median :0.0
0000
 Mean   :0.0008588   Mean   :0.01889   Mean   :0.02525   Mean   :0.0
1563
 3rd Qu.:0.0000000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0
0000
 Max.   :3.0000000   Max.   :6.00000   Max.   :1.00000   Max.   :6.0
0000
   X.PBYSTAND         X.AWAPART        X.AWABEDR          X.AWALAND
 Min.   :0.00000   Min.   :0.000   Min.   :0.00000   Min.   :0.00000
 1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.00000   1st Qu.:0.00000
 Median :0.00000   Median :0.000   Median :0.00000   Median :0.00000
 Mean   :0.04758   Mean   :0.403   Mean   :0.01477   Mean   :0.02061
 3rd Qu.:0.00000   3rd Qu.:1.000   3rd Qu.:0.00000   3rd Qu.:0.00000
 Max.   :5.00000   Max.   :2.000   Max.   :5.00000   Max.   :1.00000
   X.APERSAUT        X.ABESAUT          X.AMOTSCO          X.AVRAAUT
 Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0000
00
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000
00
 Median :1.0000   Median :0.00000   Median :0.00000   Median :0.0000
00
 Mean   :0.5622   Mean   :0.01048   Mean   :0.04105   Mean   :0.0022
33
 3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0000
00
 Max.   :7.0000   Max.   :4.00000   Max.   :8.00000   Max.   :3.0000
00
   X.AAANHANG         X.ATRACTOR          X.AWERKT           X.ABROM
 Min.   :0.00000   Min.   :0.00000   Min.   :0.000000   Min.   :0.00
000
 1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.00
000
 Median :0.00000   Median :0.00000   Median :0.000000   Median :0.00
000
 Mean   :0.01254   Mean   :0.03367   Mean   :0.006183   Mean   :0.07
042
 3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.00
000
 Max.   :3.00000   Max.   :4.00000   Max.   :6.000000   Max.   :2.00
000
    X.ALEVEN          X.APERSONG          X.AGEZONG          X.AWAOREG
 Min.   :0.00000   Min.   :0.000000   Min.   :0.000000   Min.   :0.0
00000
 1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.000000   1st Qu.:0.0
00000
 Median :0.00000   Median :0.000000   Median :0.000000   Median :0.0
00000
 Mean   :0.07661   Mean   :0.005325   Mean   :0.006527   Mean   :0.0
04638
 3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.000000   3rd Qu.:0.0
00000
 Max.   :8.00000   Max.   :1.000000   Max.   :1.000000   Max.   :2.0
```

```
00000
    X.ABRAND         X.AZEILPL           X.APLEZIER          X.AFIETS
 Min.   :0.0000   Min.   :0.0000000   Min.   :0.000000   Min.   :0.0
0000
 1st Qu.:0.0000   1st Qu.:0.0000000   1st Qu.:0.000000   1st Qu.:0.0
0000
 Median :1.0000   Median :0.0000000   Median :0.000000   Median :0.0
0000
 Mean   :0.5701   Mean   :0.0005153   Mean   :0.006012   Mean   :0.0
3178
 3rd Qu.:1.0000   3rd Qu.:0.0000000   3rd Qu.:0.000000   3rd Qu.:0.0
0000
 Max.   :7.0000   Max.   :1.0000000   Max.   :2.000000   Max.   :3.0
0000
   X.AINBOED          X.ABYSTAND         X.CARAVAN
 Min.   :0.000000   Min.   :0.00000   Min.   :0.00000
 1st Qu.:0.000000   1st Qu.:0.00000   1st Qu.:0.00000
 Median :0.000000   Median :0.00000   Median :0.00000
 Mean   :0.007901   Mean   :0.01426   Mean   :0.05977
 3rd Qu.:0.000000   3rd Qu.:0.00000   3rd Qu.:0.00000
 Max.   :2.000000   Max.   :2.00000   Max.   :1.00000
```

From the summary we can make out that almost all the features have nominal values. let us have a look at the structure of the dataset.

In [9]:

```
str(train_data)
```

```
'data.frame':   5822 obs. of  86 variables:
 $ X.MOSTYPE : int  33 37 37 9 40 23 39 33 33 11 ...
 $ X.MAANTHUI: int  1 1 1 1 1 1 2 1 1 2 ...
 $ X.MGEMOMV : int  3 2 2 3 4 2 3 2 2 3 ...
 $ X.MGEMLEEF: int  2 2 2 3 2 1 2 3 4 3 ...
 $ X.MOSHOOFD: int  8 8 8 3 10 5 9 8 8 3 ...
 $ X.MGODRK  : int  0 1 0 2 1 0 2 0 0 3 ...
 $ X.MGODPR  : int  5 4 4 3 4 5 2 7 1 5 ...
 $ X.MGODOV  : int  1 1 2 2 1 0 0 0 3 0 ...
 $ X.MGODGE  : int  3 4 4 4 4 5 5 2 6 2 ...
 $ X.MRELGE  : int  7 6 3 5 7 0 7 7 6 7 ...
 $ X.MRELSA  : int  0 2 2 2 1 6 2 2 0 0 ...
 $ X.MRELOV  : int  2 2 4 2 2 3 0 0 3 2 ...
 $ X.MFALLEEN: int  1 0 4 2 2 3 0 0 3 2 ...
 $ X.MFGEKIND: int  2 4 4 3 4 5 3 5 3 2 ...
 $ X.MFWEKIND: int  6 5 2 4 4 2 6 4 3 6 ...
 $ X.MOPLHOOG: int  1 0 0 3 5 0 0 0 0 0 ...
 $ X.MOPLMIDD: int  2 5 5 4 4 5 4 3 1 4 ...
 $ X.MOPLLAAG: int  7 4 4 2 0 4 5 6 8 5 ...
 $ X.MBERHOOG: int  1 0 0 4 0 2 0 2 1 2 ...
 $ X.MBERZELF: int  0 0 0 0 5 0 0 0 1 0 ...
 $ X.MBERBOER: int  1 0 0 0 4 0 0 0 0 0 ...
 $ X.MBERMIDD: int  2 5 7 3 0 4 4 2 1 3 ...
 $ X.MBERARBG: int  5 0 0 1 0 2 1 5 8 3 ...
 $ X.MBERARBO: int  2 4 2 2 0 2 5 2 1 3 ...
 $ X.MSKA    : int  1 0 0 3 9 2 0 2 1 1 ...
 $ X.MSKB    : int  1 2 5 2 0 2 1 1 1 2 ...
 $ X.MSKB.1  : int  2 3 0 1 0 2 4 2 0 1 ...
 $ X.MSKC    : int  6 5 4 4 0 4 5 5 8 4 ...
 $ X.MSKD    : int  1 0 0 0 0 2 0 2 1 2 ...
 $ X.MHHUUR  : int  1 2 7 5 4 9 6 0 9 0 ...
 $ X.MHKOOP  : int  8 7 2 4 5 0 3 9 0 9 ...
 $ X.MAUT    : int  8 7 7 9 6 5 8 4 5 6 ...
 $ X.MAUT.1  : int  0 1 0 0 2 3 0 4 2 1 ...
 $ X.MAUT.2  : int  1 2 2 0 1 3 1 2 3 2 ...
 $ X.MZFONDS : int  8 6 9 7 5 9 9 6 7 6 ...
 $ X.MZPART  : int  1 3 0 2 4 0 0 3 2 3 ...
 $ X.MINKM   : int  0 2 4 1 0 5 4 2 7 2 ...
 $ X.MINK    : int  4 0 5 5 0 2 3 5 2 3 ...
 $ X.MINK.1  : int  5 5 0 3 9 3 3 3 1 3 ...
 $ X.MINK.2  : int  0 2 0 0 0 0 0 0 0 1 ...
 $ X.MINK.3  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.MINKGEM : int  4 5 3 4 6 3 3 3 2 4 ...
 $ X.MKOOPKLA: int  3 4 4 4 3 3 5 3 3 7 ...
 $ X.PWAPART : int  0 2 2 0 0 0 0 0 0 2 ...
 $ X.PWABEDR : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PWALAND : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PPERSAUT: int  6 0 6 6 0 6 6 0 5 0 ...
 $ X.PBESAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PMOTSCO : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PVRAAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PAANHANG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PTRACTOR: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PWERKT  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PBROM   : int  0 0 0 0 0 0 0 3 0 0 ...
 $ X.PLEVEN  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PPERSONG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PGEZONG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PWAOREG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PBRAND  : int  5 2 2 2 6 0 0 0 0 3 ...
 $ X.PZEILPL : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
 $ X.PPLEZIER: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PFIETS  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PINBOED : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.PBYSTAND: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AWAPART : int  0 2 1 0 0 0 0 0 0 1 ...
 $ X.AWABEDR : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AWALAND : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.APERSAUT: int  1 0 1 1 0 1 1 0 1 0 ...
 $ X.ABESAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AMOTSCO : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AVRAAUT : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AAANHANG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.ATRACTOR: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AWERKT  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.ABROM   : int  0 0 0 0 0 0 0 1 0 0 ...
 $ X.ALEVEN  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.APERSONG: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AGEZONG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AWAOREG : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.ABRAND  : int  1 1 1 1 1 0 0 0 0 1 ...
 $ X.AZEILPL : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.APLEZIER: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AFIETS  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.AINBOED : int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.ABYSTAND: int  0 0 0 0 0 0 0 0 0 0 ...
 $ X.CARAVAN : int  0 0 0 0 0 0 0 0 0 0 ...
```

We can confirm from the dataset that all the variables are of integer type. we will have to perform further analysis on what needs to be done with these variables.

Let us examine how the the response variable, Caravan in distributed in the dataset.

In [10]:

```
table(train_data$X.CARAVAN)
```

```
   0    1 
5474  348 
```

We can see from the cross tabulation that most of the customers have not opted for Caravan policy, whereas only a miniscule percentage of people have opted for Caravan policy.

In [11]:

```
print(paste("Percentage of customers who did not opt for a caravan policy is : "
, (5474/5822)*100))
print(paste("Percentage of customers who did opt for a caravan policy is : " , (
348/5822)*100))
```

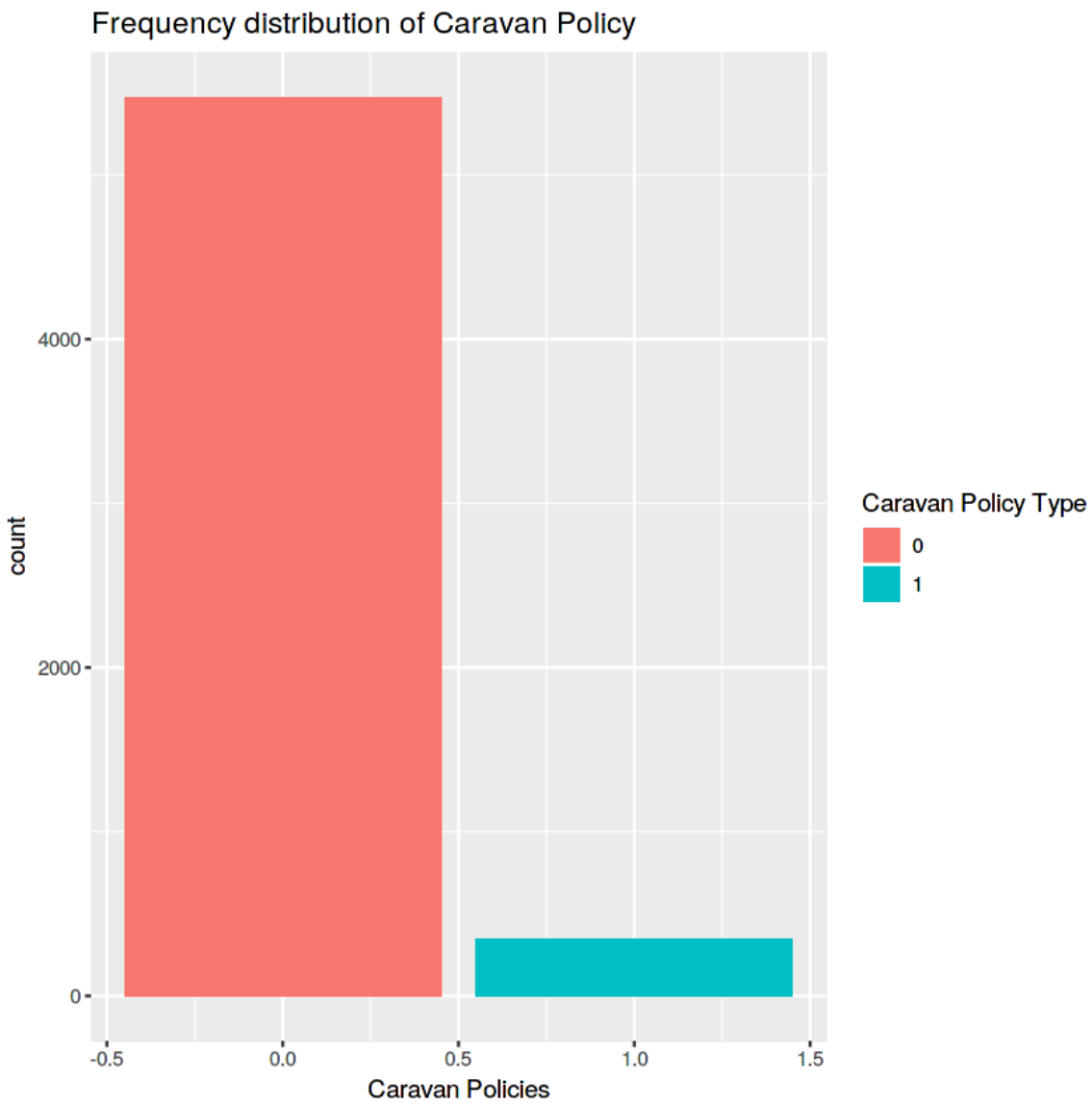
```
[1] "Percentage of customers who did not opt for a caravan policy is
:  94.0226726210924"
[1] "Percentage of customers who did opt for a caravan policy is :
5.97732737890759"
```

To quantify in percentage terms we can see that nearly 95% of the customers have not opted for caravan policy whereas, only 6% of the customers have opted for caravan policy.The same can be observed in the distribution of the caravan policy

In [12]:

```
exp <- ggplot(train_data,aes(fill = as.factor(X.CARAVAN)))+geom_histogram(aes(x
= X.CARAVAN,label="caravan policy type"),stat = "count")+ggtitle("Frequency dist
ribution of Caravan Policy")
freq_cnt <- exp + xlab("Caravan Policies") + labs(fill = "Caravan Policy Type")
freq_cnt
```

```
Warning message:
"Ignoring unknown parameters: binwidth, bins, pad"Warning message:
"Ignoring unknown aesthetics: label"
```



3. Exploratory Data Analysis

As part of Data exploration we will be performing various plots for the each of the features in the dataset. The plots will include the distribution of each of the feature, the distribution with respect to response variable, box plots and scatter plots. A footnote to conclude the suitability of the varaible as one of the strong predictors for the model.

Before we step into performing Exploration. As part of the flow chart/approach or methodology, we can first perform removal Zero/Near Zero Variance variables. The whole point of performong Exploratory data analysis is to find the features that are highly correlated or would prove to be a good predictor. Removal of Zero/nearZeroVariance features would reduce the number of features and would leave us with a smaller subset of features to select from. In modeling, the features that do not show much variance are considered to contribute less towards the predictability of the model and therefore, it would make more sense to get rid of these variables and then proceed with further analysis.

3.1 Removal of Zero/nearZeroVariance Features

Let's make use of the nearZeroVar function which will take the entire dataset and returns the index of the features which exhibit zero/near zero variance. In this step we will get rid of these features. let us make a copy of the train dataset

In [13]:

```
train_data_cop <- data.frame(train_data)
```

In [14]:

```
nzv3 <- nearZeroVar(train_data_cop)
nzv3
```

```
45  46  48  49  50  51  52  53  54  55  56  57  58  60  61  62  63
64  66  67  69  70  71  72  73  74  76  77  78  79  81  82  83  84
85
```

In [15]:

```
train_data_cop <- train_data_cop[,-nzv3]
ncol(train_data_cop)
```

```
51
```

We now perform EDA on the remaining features. we will write a function that creates a set of plots for all the features and also provide a footnote describing the characteristics of the feature.

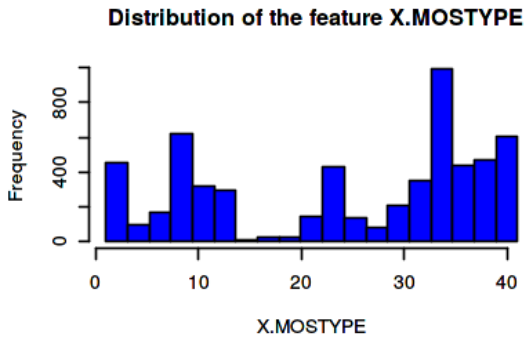
In [16]:

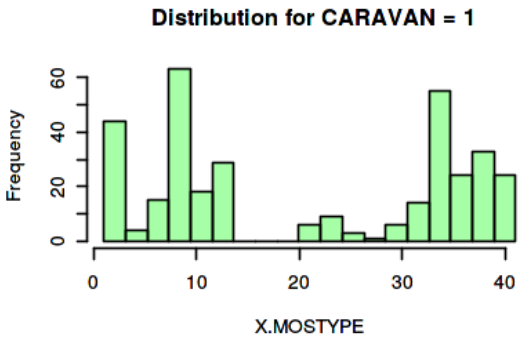
```
# Define a two-row by two-column plotting area.

exploreFeature <- function(train_data,feature){
    par(mfrow = c(3, 2.5))
    print(feature)
    # Plot a histogram and box plot for each of the predictors,
    # by response ("good" or "bad").
    #distribution of the variable
    min_d <- min(train_data[ , feature])
    max_d <- max(train_data[ , feature])
    b <- seq(min_d, max_d, length.out = 20)
    #ggplot(train_data)+geom_histogram(aes(x = train_data[,feature]),stat = "cou
nt")+ggtitle(paste("Frequency distribution of feature ",feature))
    #ggplot(train_data,aes(fill = as.factor(X.CARAVAN)))+geom_histogram(aes(x =
 train_data[,feature],label=paste("distribution for feature ",feature)),stat =
 "count")+ggtitle(paste("Frequency distribution of feature ",feature))

    #ggplot(train_data)+geom_histogram(data = train_data,aes(x = train_data$feat
ure),stat = "count")
     hist(train_data[, feature],breaks = b,xlab = feature, main=paste(" Distribu
tion of the feature",feature),col="blue")
     hist(train_data[ , feature][train_data$X.CARAVAN == 1], col = rgb(0, 1, 0,
0.35), breaks = b,
             main = paste(" Distribution for CARAVAN = 1" ,sep = ""), xlab = fea
ture)
     hist(train_data[ , feature][train_data$X.CARAVAN == 0], xlab = feature, col
= rgb(1, 0, 0, 0.35), breaks = b,main=paste("Distribution for CARAVAN = 0"),
             add = FALSE,)
     boxplot(train_data[ , feature] ~ train_data$X.CARAVAN, col = c(rgb(1, 0, 0,
0.35),rgb(0, 1, 0, 0.35)),notch = TRUE,varwidth = TRUE,main = paste("Boxplot: ",
feature, sep = ""))
     plot(train_data$X.CARAVAN,train_data[ , feature],xlab="CARAVAN", ylab=featu
re, pch=19,col = "blue")
     chisq.test(train_data[,feature],train_data_cop$X.CARAVAN)
}
```

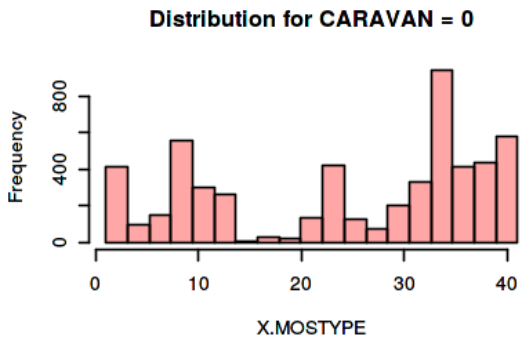
In [17]:

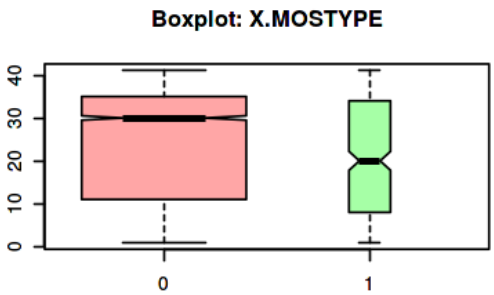
```
exploreFeature(train_data_cop,colnames(train_data_cop[1]))
```

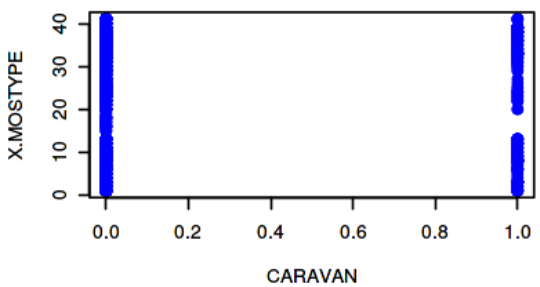
```
[1] "X.MOSTYPE"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 124.81, df = 39, p-value = 6.527e-11
```









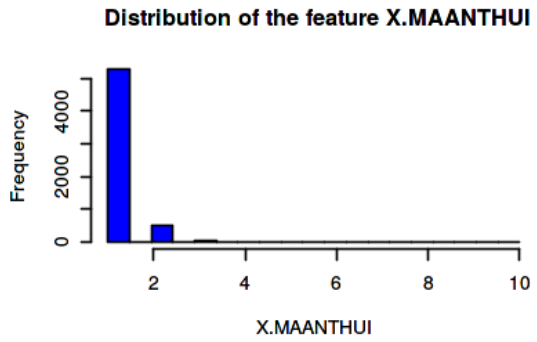


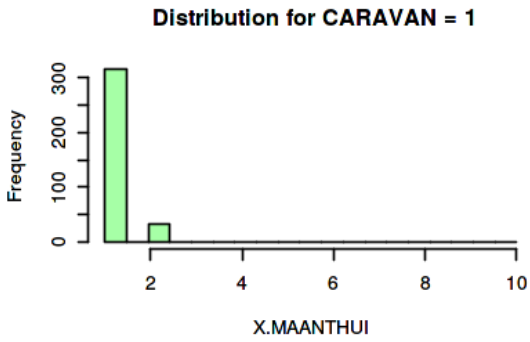
*The feature under inspection here is **Customer Subtype**. The distribution and the varaible description indicates that this is a categorical variable. The distribution indicates the the number of customers for each of the classes. Although the distribution with respect to CARAVAN does not change so much, we can see from the box plot that the median class from CARAVAN =0 is 30 and CARAVAN=1 is 20. Customers belonging to subtype 8(middle class) and 33 are most likely to buy the Caravan policies. We performed chi-square hypothesis test which indicates a p-value very close to 0. which means we have strong evidence against the null hypothesis(The feature is independent of response variable). This will be chosen as one of the predictors for our model.*

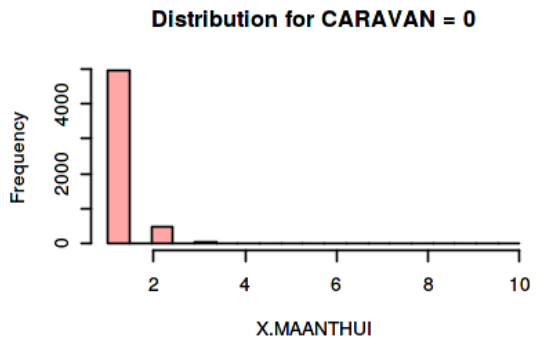
In [18]:

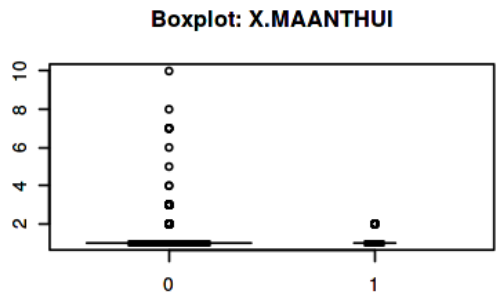
```
exploreFeature(train_data_cop,colnames(train_data_cop[2]))
```

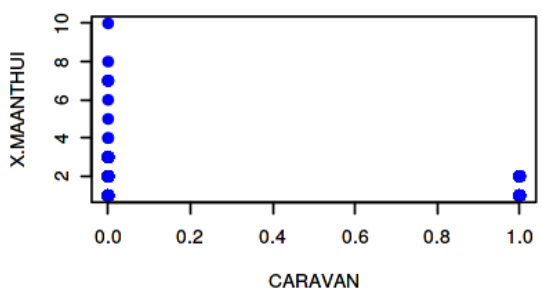
```
[1] "X.MAANTHUI"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 3.4579, df = 8, p-value = 0.9024
```









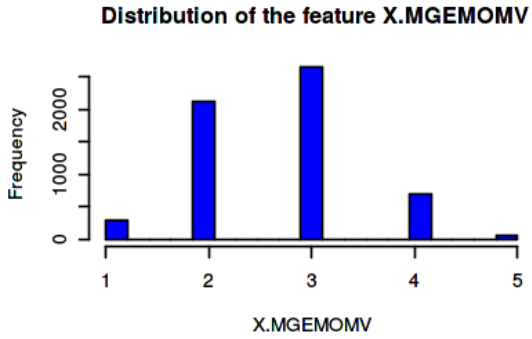


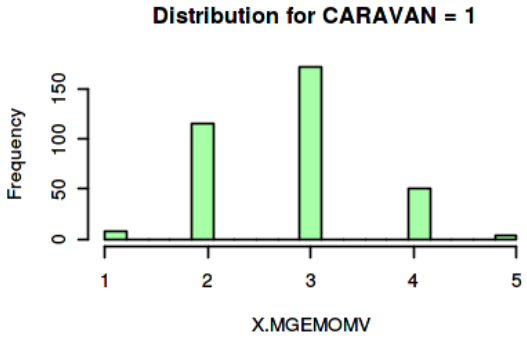
*The feature under inspection here is **Number of houses**. The distribution doesn't give much information about the interpretability of the response varaible. Therefore, we can disregard this variable as one of the predictors*

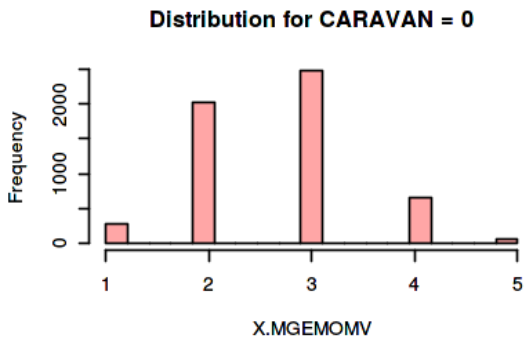
In [19]:

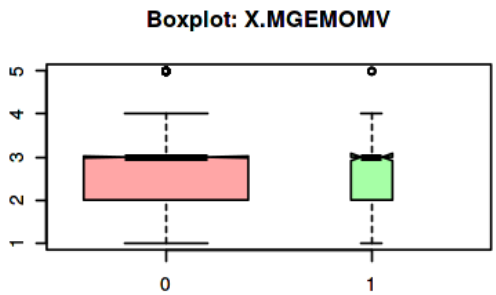
```
exploreFeature(train_data_cop,colnames(train_data_cop[3]))
```

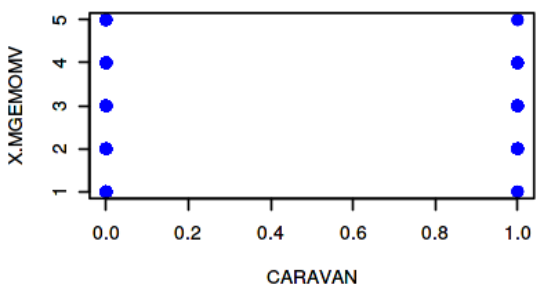
```
[1] "X.MGEMOMV"

Warning message in bxp(structure(list(stats = structure(c(1, 2, 3,
3, 4, 1, 2, 3, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 9.3252, df = 4, p-value = 0.05347
```









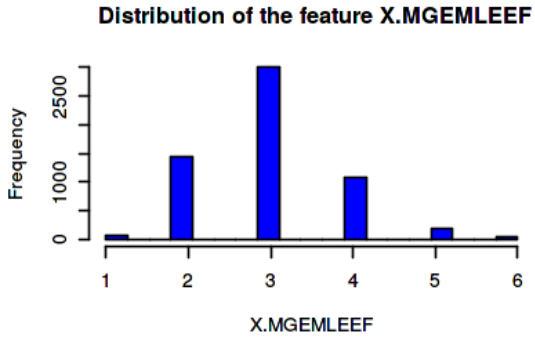


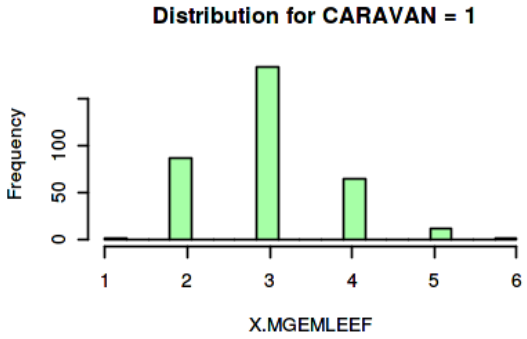
*The feature under inspection here is **Avg size household**. The values for this feature as can be seen from the description and distribution plot varies from 1-6. However, the classes for each of the response variable seem to be equally distributed and therefore, we can disregard this as a good predictor.*

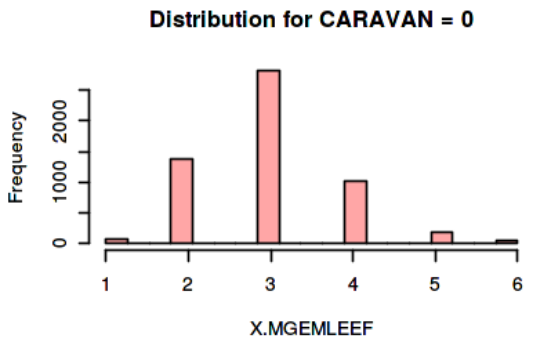
In [20]:

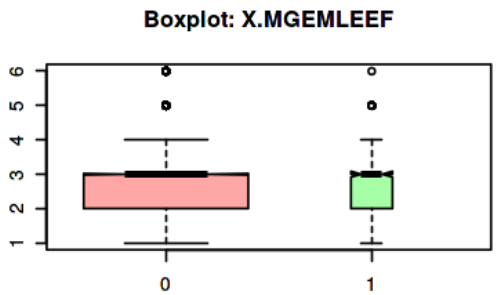
```
exploreFeature(train_data_cop,colnames(train_data_cop[4]))
```

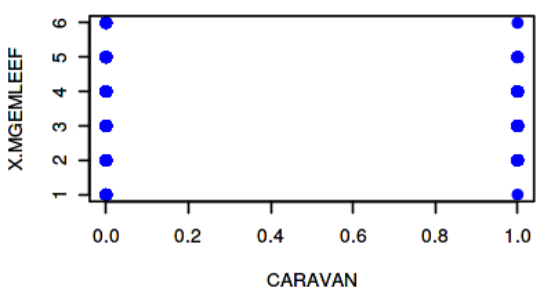
```
[1] "X.MGEMLEEF"

Warning message in bxp(structure(list(stats = structure(c(1, 2, 3,
3, 4, 1, 2, 3, :
“some notches went outside hinges ('box'): maybe set notch=FALSE”War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
“Chi-squared approximation may be incorrect”

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 3.2919, df = 5, p-value = 0.6551
```









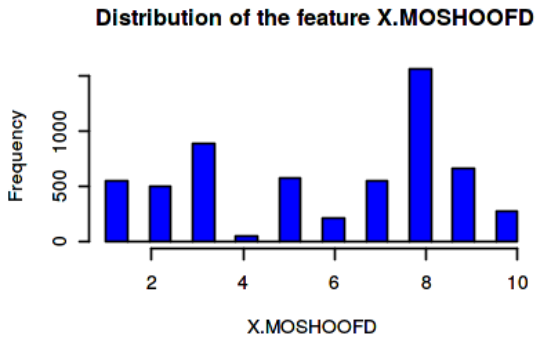


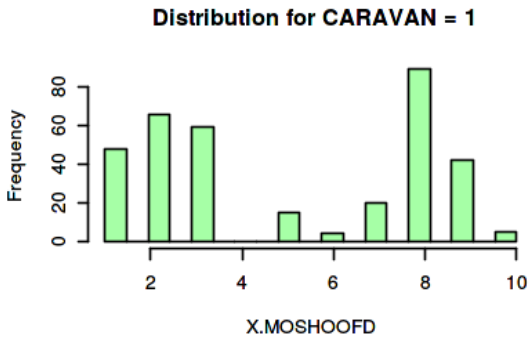
*The feature under inspection here is **Avg age**. This feature shows the average age group of customers who tend to buy Caravn policy. But from the distribution of the feature with respect to Caravan policy, there is no clear demarcation among different sub groups . the same can be observed in Box-plots and the scatter plot as well. Therefore, we can disregard this varaible as one of the predictors.*

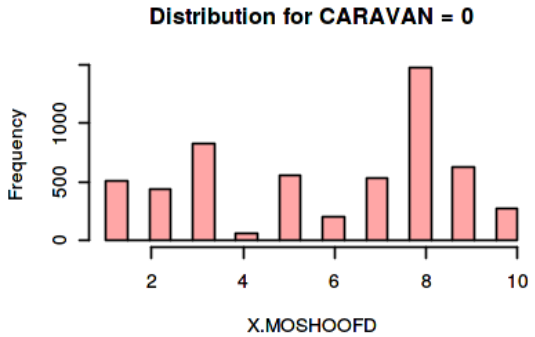
In [21]:

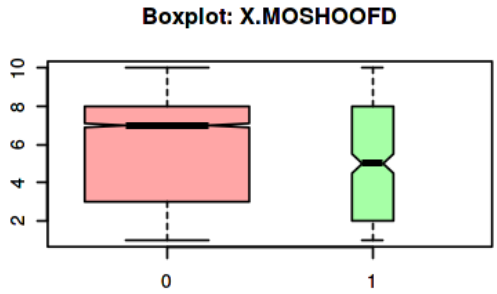
```
exploreFeature(train_data_cop,colnames(train_data_cop[5]))
```

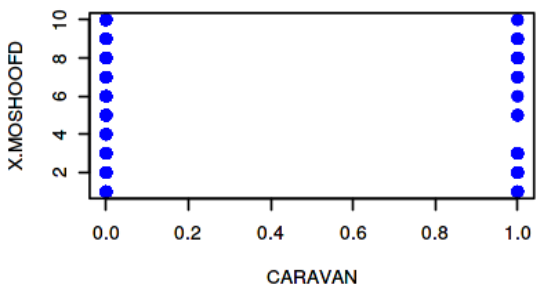
```
[1] "X.MOSHOOFD"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 88.662, df = 9, p-value = 3.02e-15
```









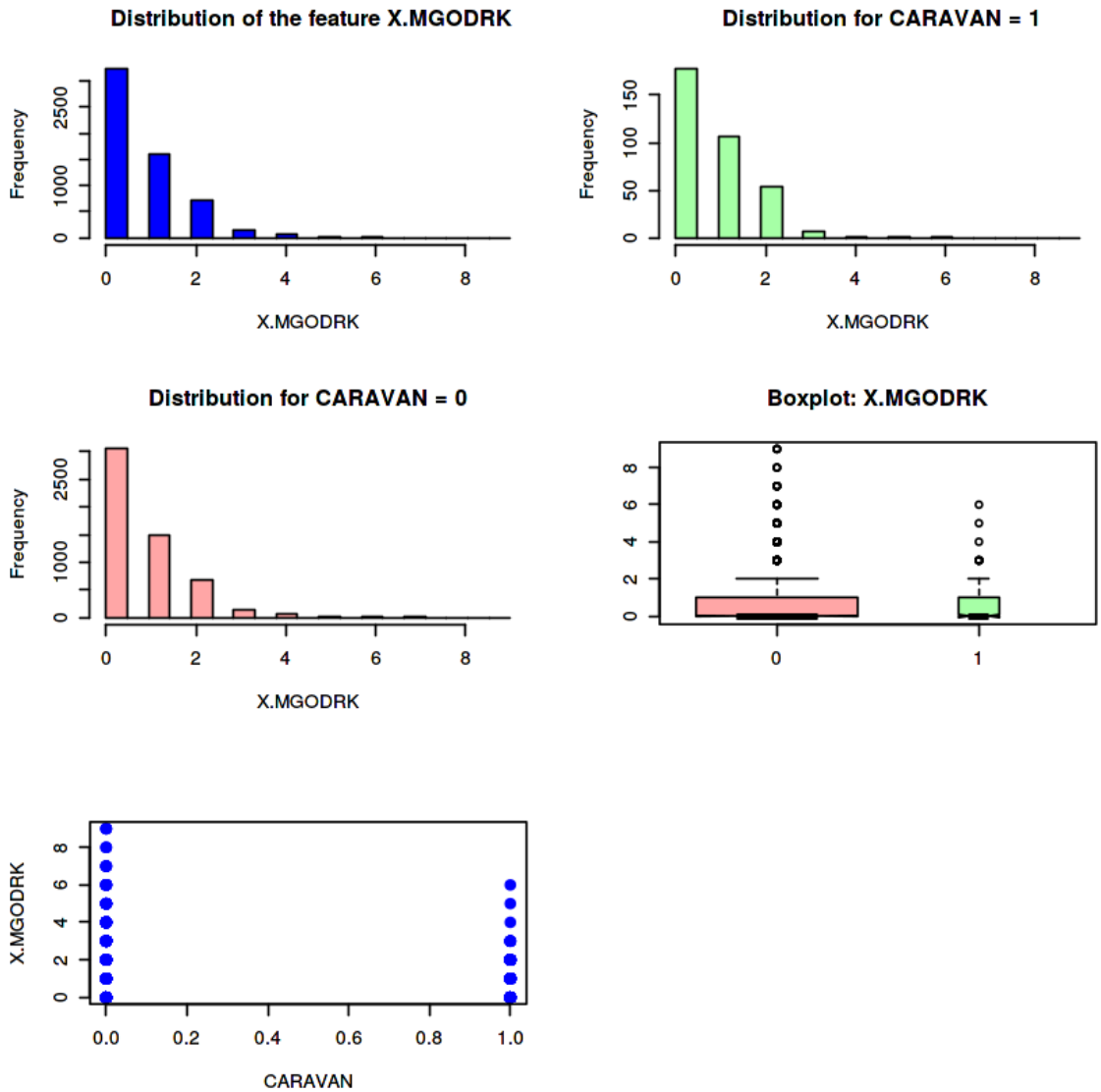


*The feature under inspection here is **Customer Main Type**. The distribution and the varaible description indicates that this is a categorical variable. The distribution indicates the the number of customers for each of the classes. Although the distribution with respect to CARAVAN does not change so much, we can see from the box plot that there is a clear demarcation among the classes. Customers belonging to subtype 8(middle class) are likely to buy the Caravan policies. We performed chi-square hypothesis test which indicates a p-value very close to 0. which means we have strong evidence against the null hypothesis(The feature is independent of response variable). This will be chosen as one of the predictors for our model.*

In [22]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[6]))
```

```
[1] "X.MGODRK"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
1, 2, 0, 0, 0, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 9.1202, df = 9, p-value = 0.4262
```

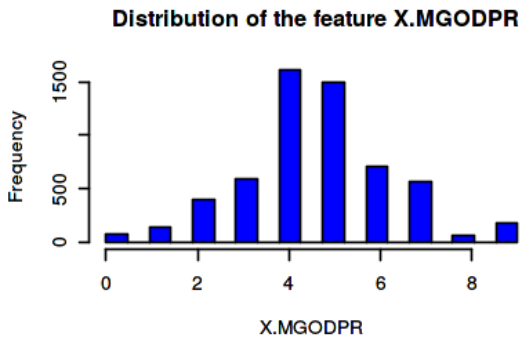


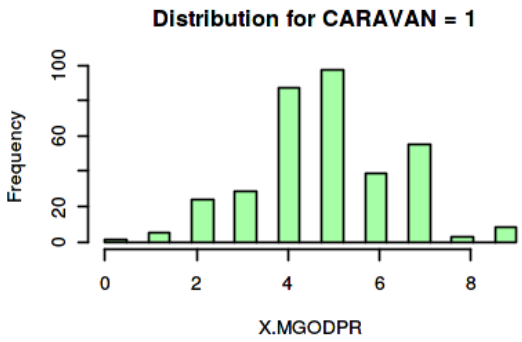
*The feature under inspection here is **Roman Catholic**.From the distribution of the feature with respect to Caravan policy, there is no clear demarcation among different sub groups . the same can be observed in Box-plots and the scatter plot as well. Therefore, we can disregard this varaible as one of the predictors.*

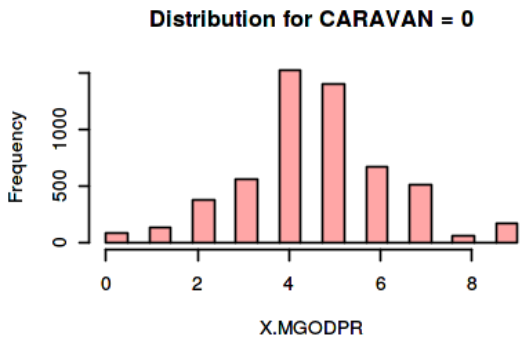
In [23]:

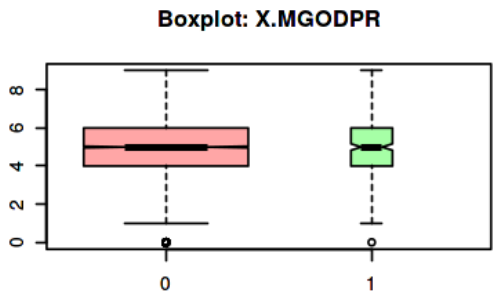
```
exploreFeature(train_data_cop,colnames(train_data_cop[7]))
```

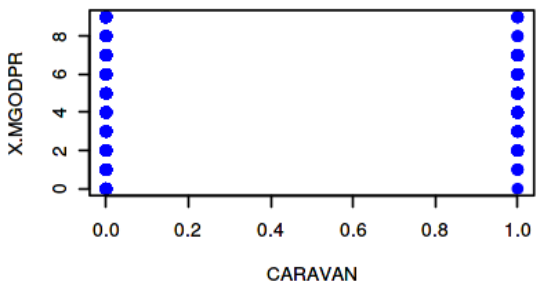
```
[1] "X.MGODPR"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 22.4, df = 9, p-value = 0.007694
```







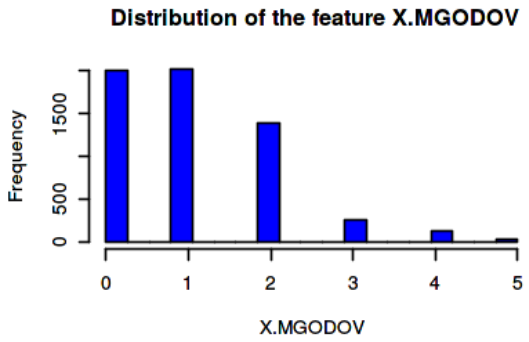


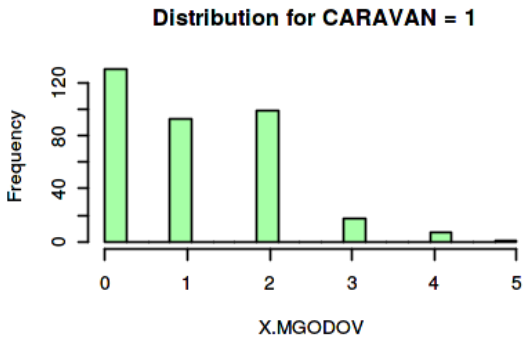


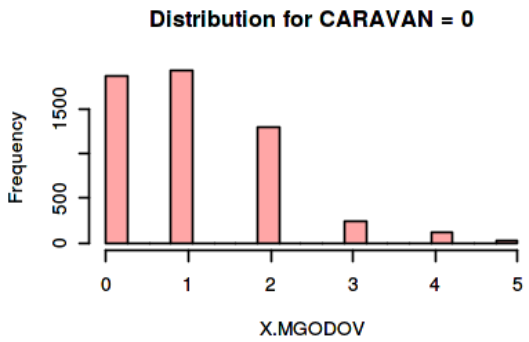
In [24]:

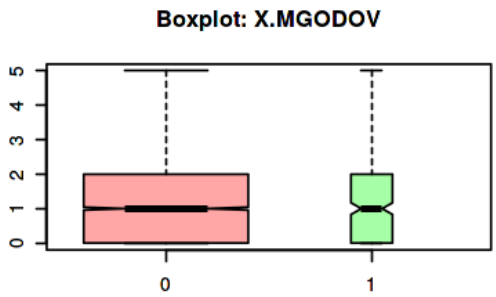
```
exploreFeature(train_data_cop,colnames(train_data_cop[8]))
```

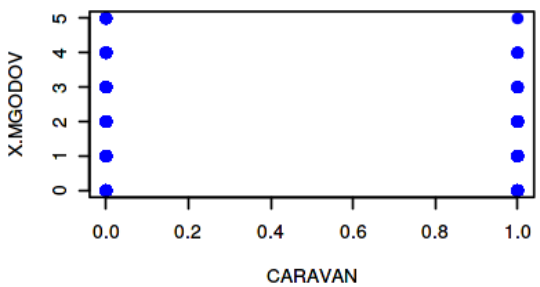
```
[1] "X.MGODOV"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 11.736, df = 5, p-value = 0.03859
```







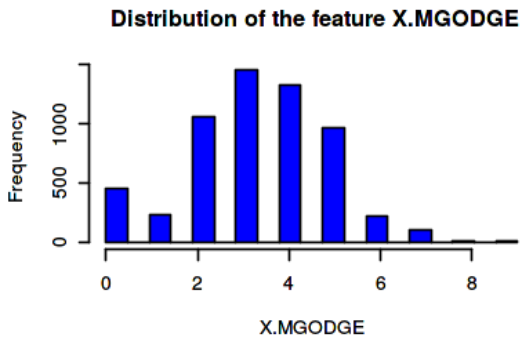


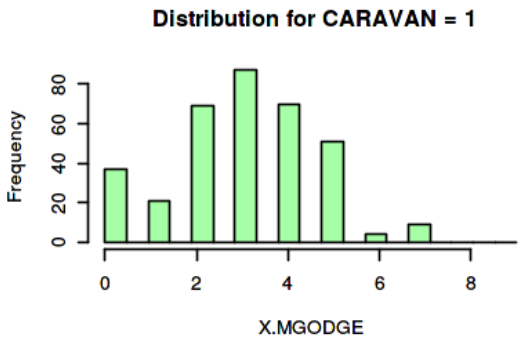


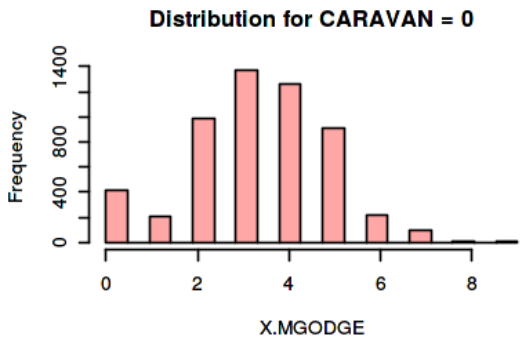
In [25]:

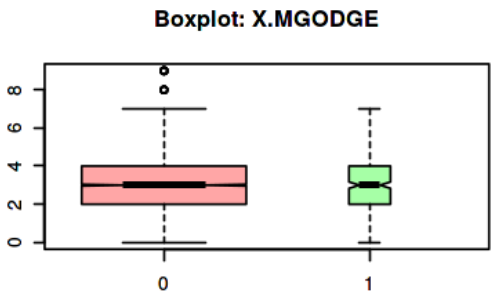
```
exploreFeature(train_data_cop,colnames(train_data_cop[9]))
```

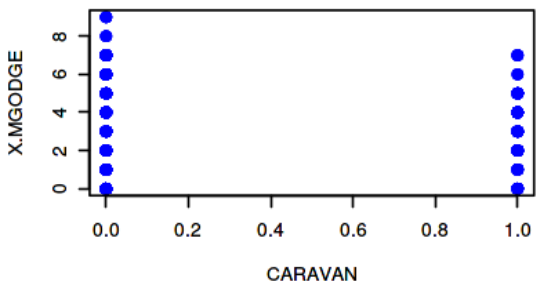
```
[1] "X.MGODGE"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 19.4, df = 9, p-value = 0.022
```







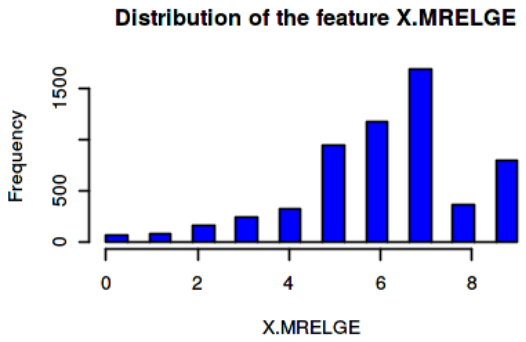


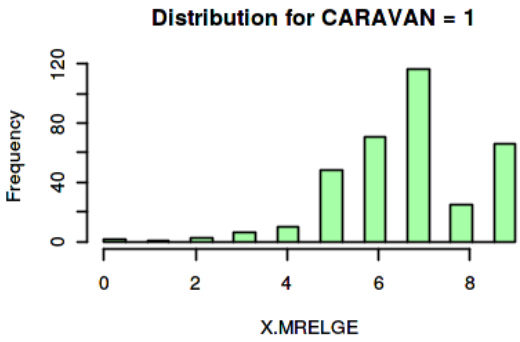


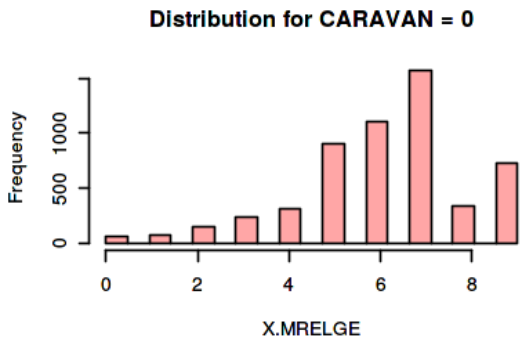
In [26]:

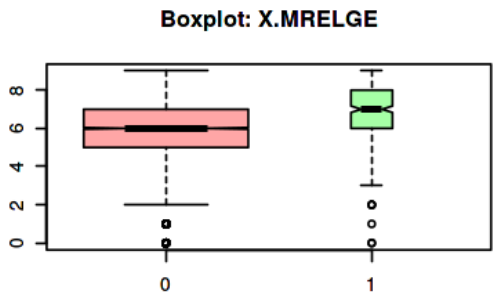
```
exploreFeature(train_data_cop,colnames(train_data_cop[10]))
```

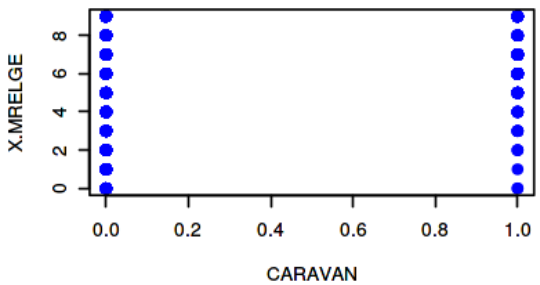
```
[1] "X.MRELGE"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 30.897, df = 9, p-value = 0.0003083
```











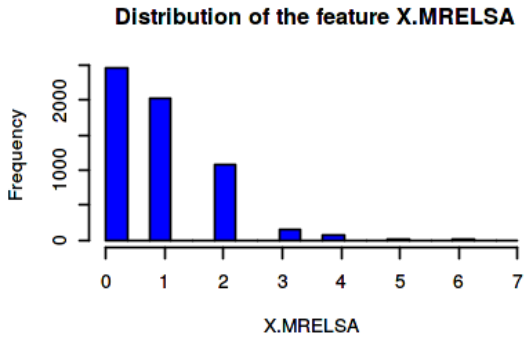
In [27]:

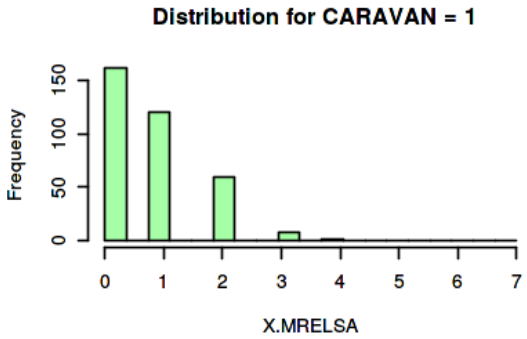
```
exploreFeature(train_data_cop,colnames(train_data_cop[11]))
```

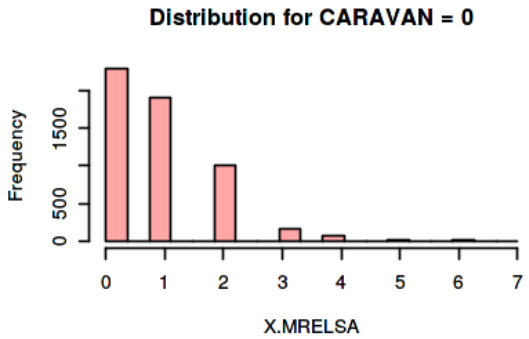
```
[1] "X.MRELSA"
```

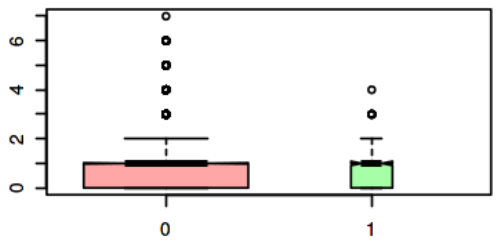
```
Warning message in bxp(structure(list(stats = structure(c(0, 0, 1,
1, 2, 0, 0, 1, :
“some notches went outside hinges ('box'): maybe set notch=FALSE”War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
“Chi-squared approximation may be incorrect”
```

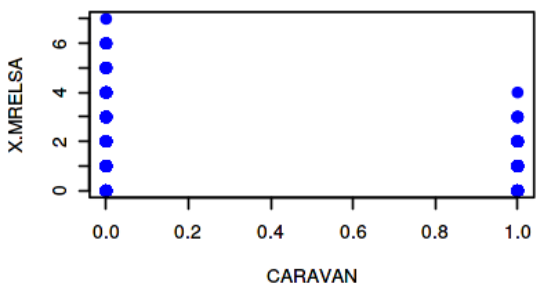
```
        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 7.834, df = 7, p-value = 0.3475
```







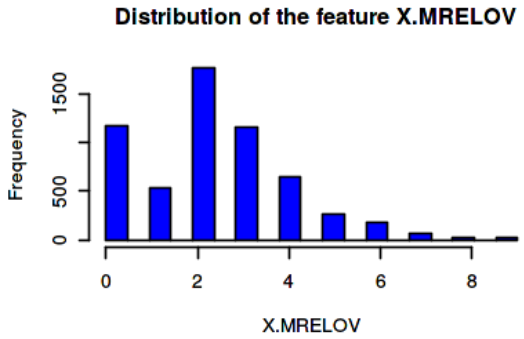


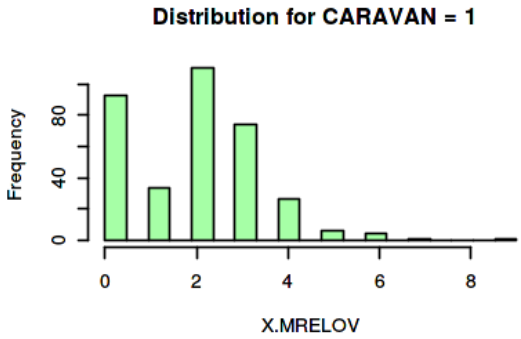


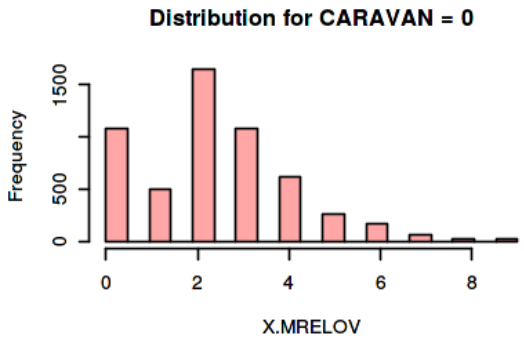
In [28]:

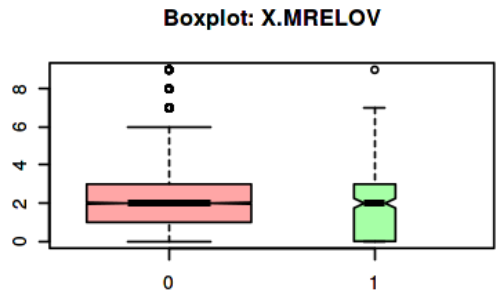
```
exploreFeature(train_data_cop,colnames(train_data_cop[12]))
```

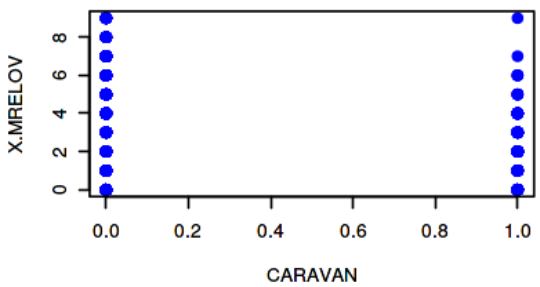
```
[1] "X.MRELOV"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 27.795, df = 9, p-value = 0.001032
```







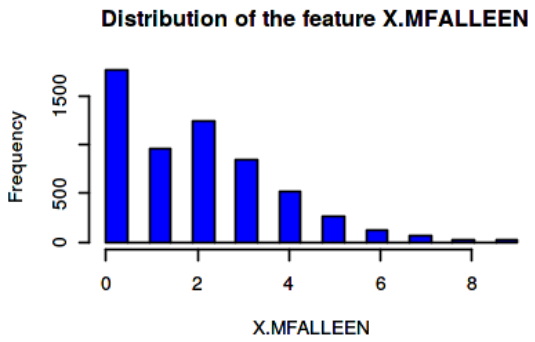


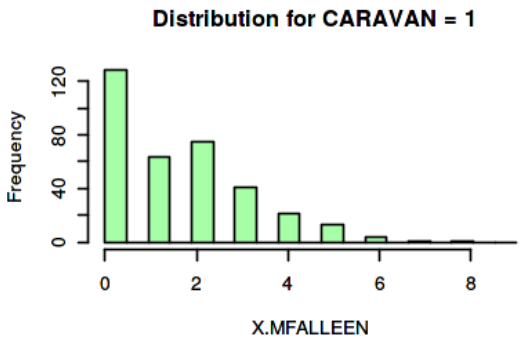


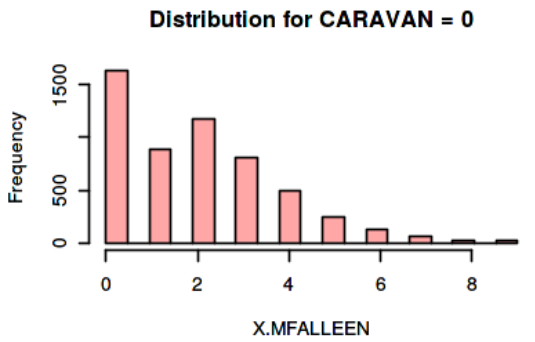
In [29]:

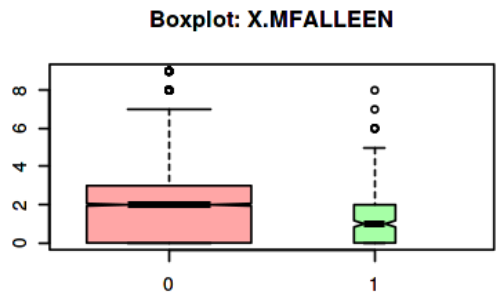
```
exploreFeature(train_data_cop,colnames(train_data_cop[13]))
```

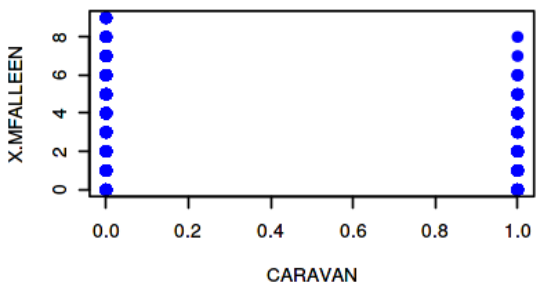
```
[1] "X.MFALLEEN"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 17.951, df = 9, p-value = 0.03574
```







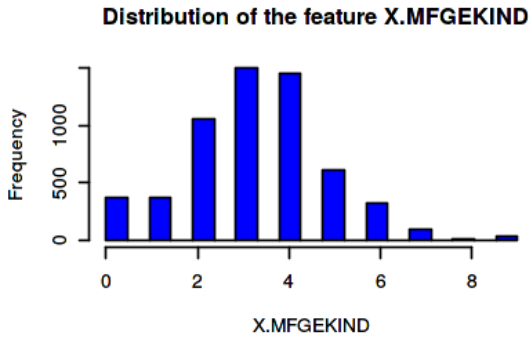


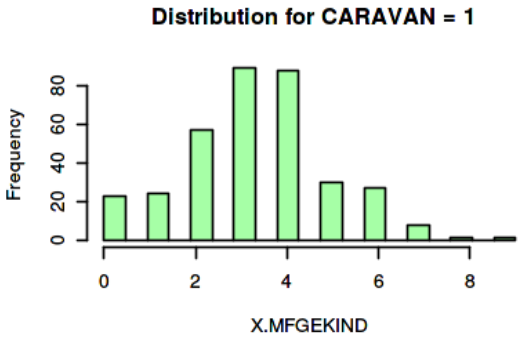


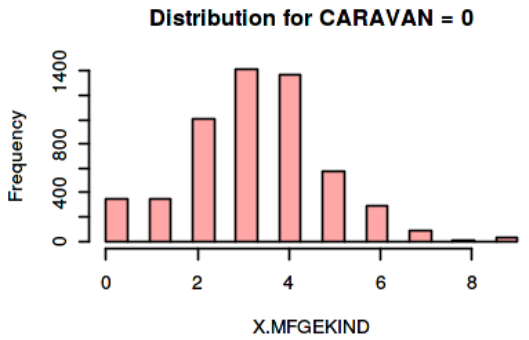
In [30]:

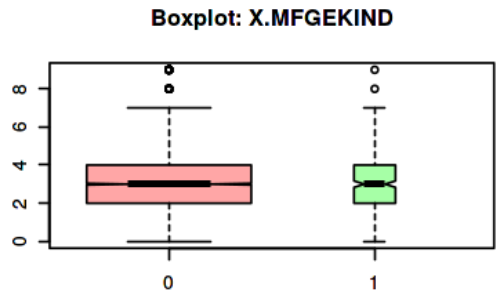
```
exploreFeature(train_data_cop,colnames(train_data_cop[14]))
```

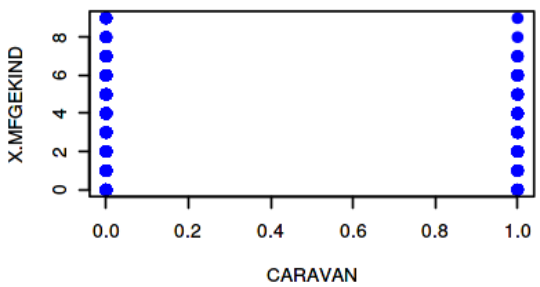
```
[1] "X.MFGEKIND"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 6.7093, df = 9, p-value = 0.6674
```







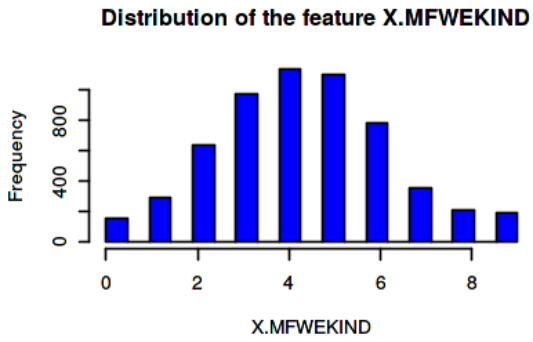


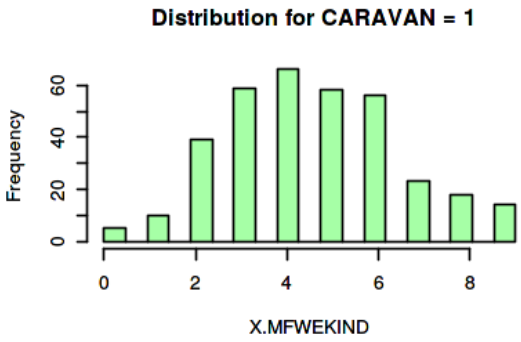


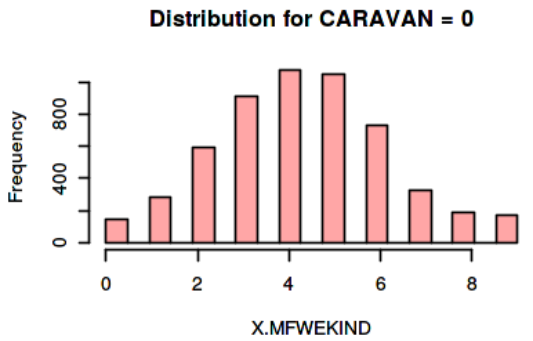
In [31]:

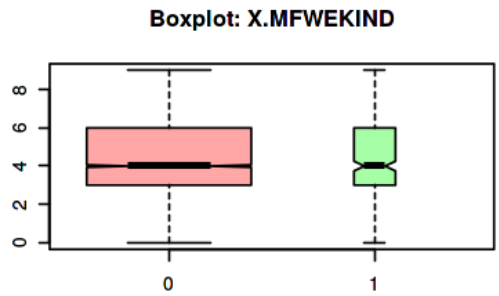
```
exploreFeature(train_data_cop,colnames(train_data_cop[15]))
```

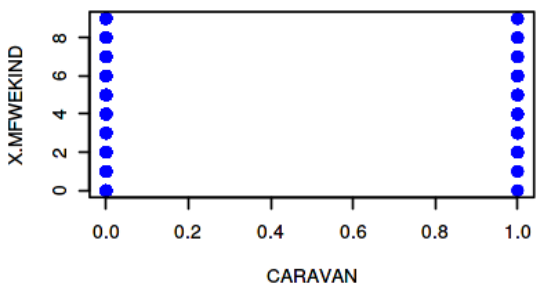
```
[1] "X.MFWEKIND"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 12.263, df = 9, p-value = 0.1989
```







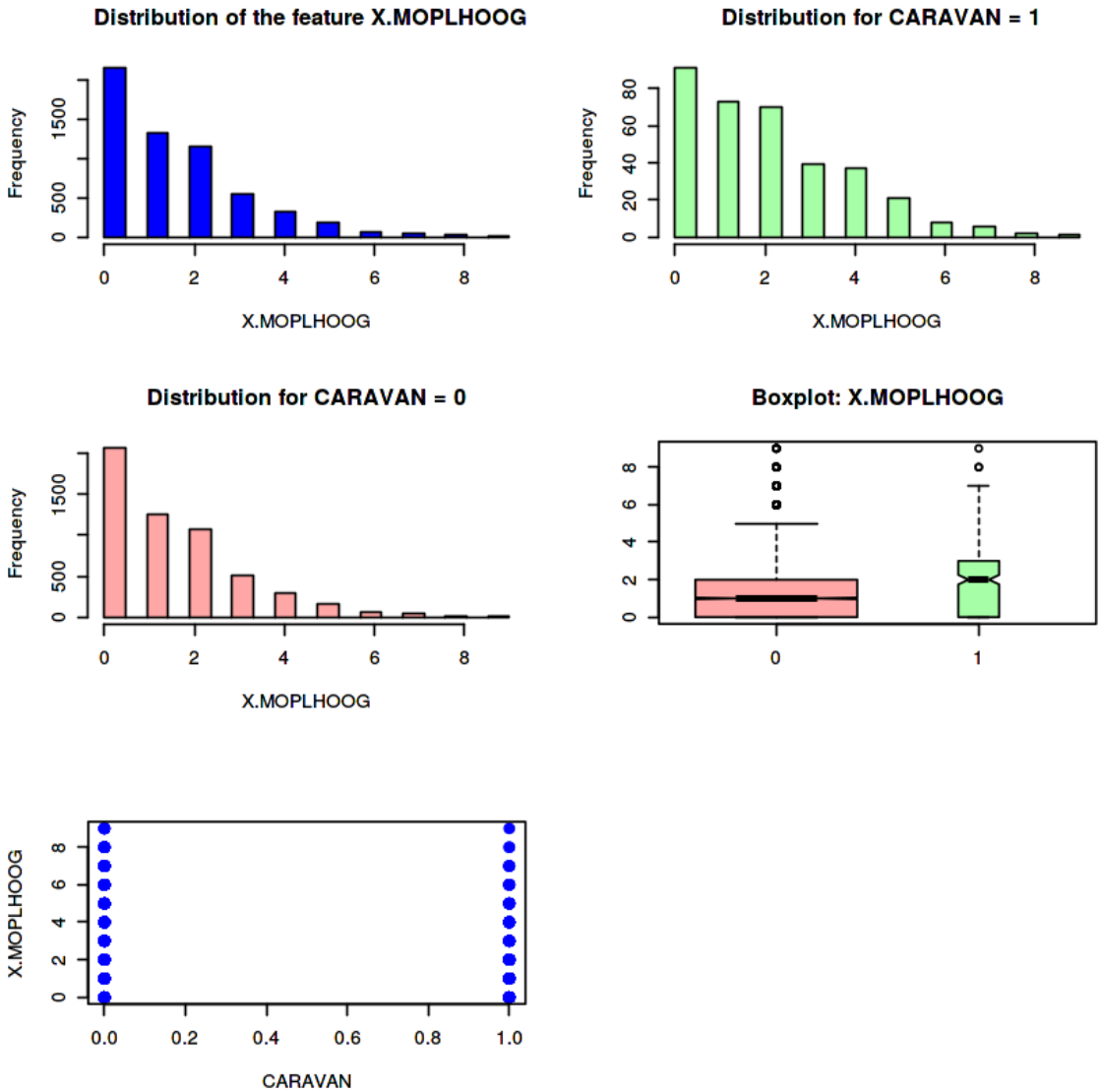




In [32]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[16]))
```

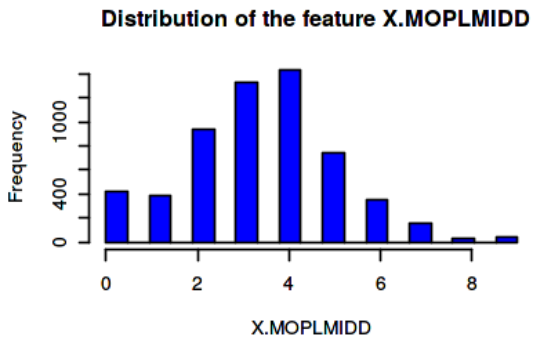
```
[1] "X.MOPLHOOG"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 47.375, df = 9, p-value = 3.342e-07
```

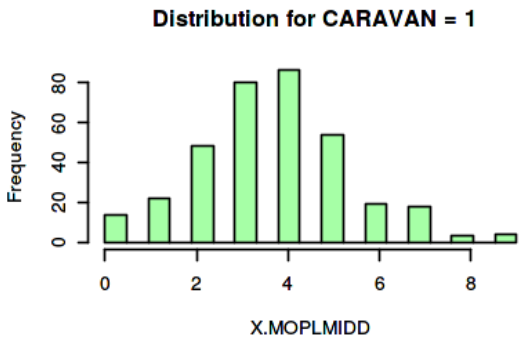


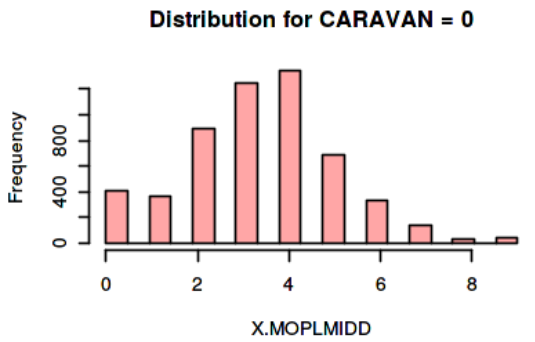
In [33]:

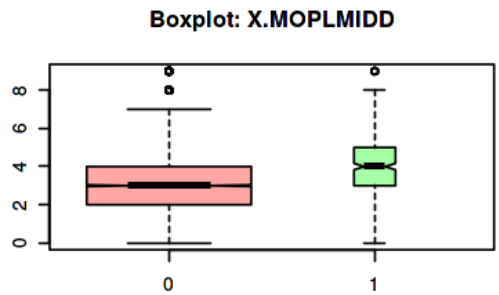
```
exploreFeature(train_data_cop,colnames(train_data_cop[17]))
```

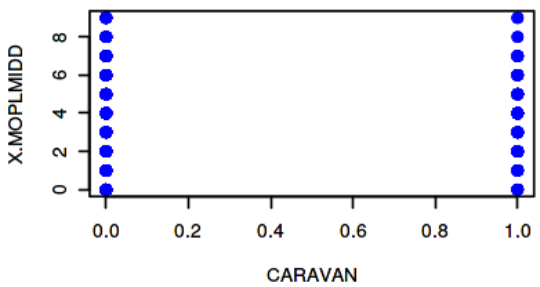
```
[1] "X.MOPLMIDD"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 18.702, df = 9, p-value = 0.02785
```







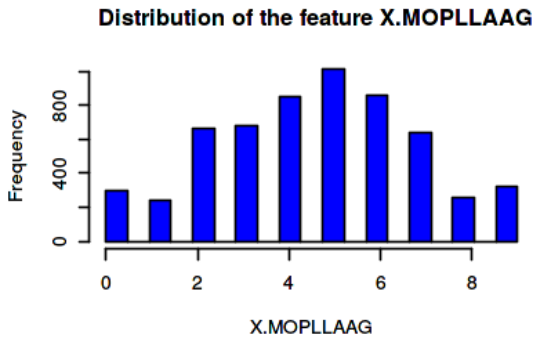


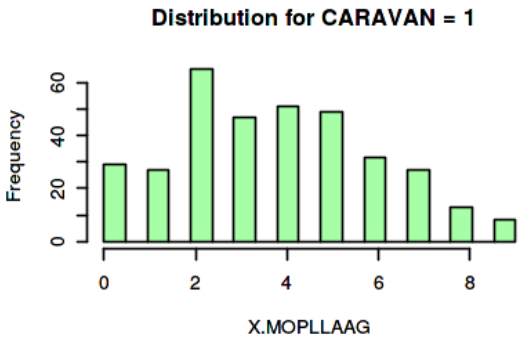


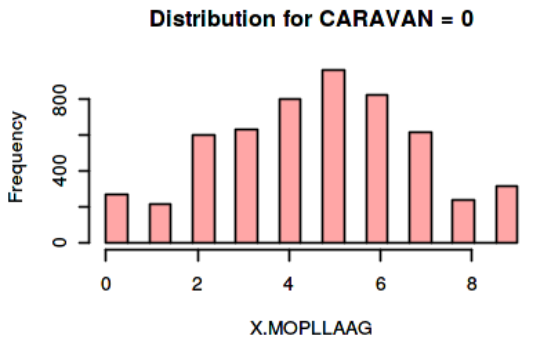
In [34]:

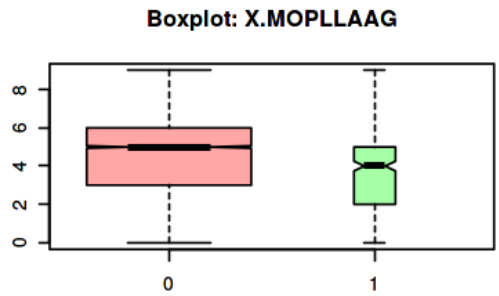
```
exploreFeature(train_data_cop,colnames(train_data_cop[18]))
```

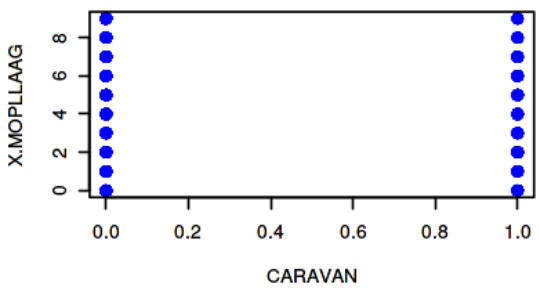
```
[1] "X.MOPLLAAG"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 57.46, df = 9, p-value = 4.125e-09
```







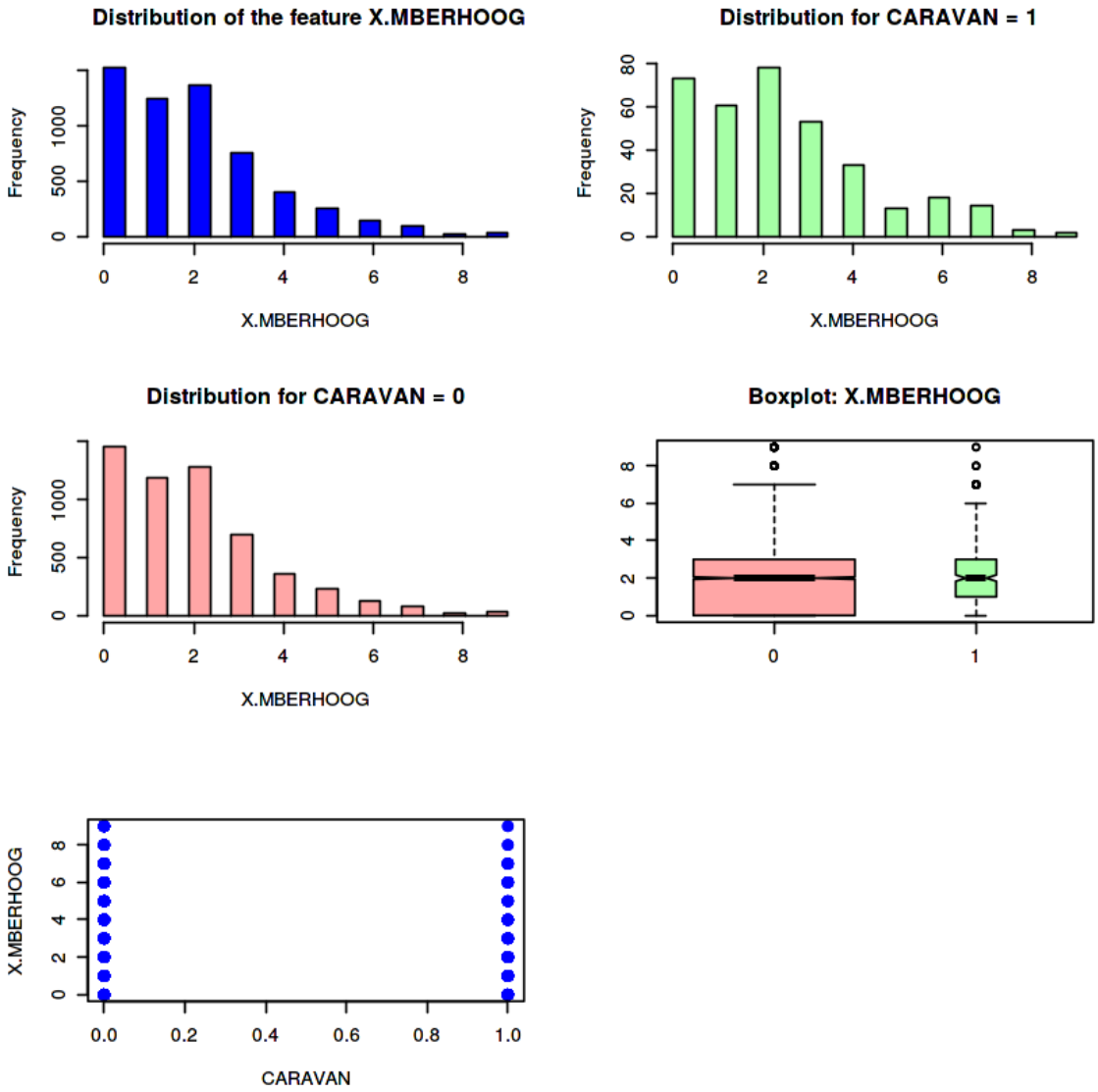




In [35]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[19]))
```

```
[1] "X.MBERHOOG"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 39.78, df = 9, p-value = 8.33e-06
```



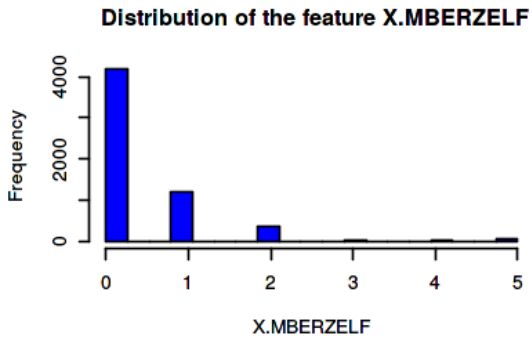
In [36]:

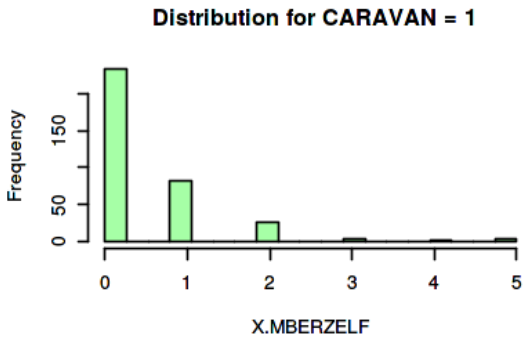
```
exploreFeature(train_data_cop,colnames(train_data_cop[20]))
```

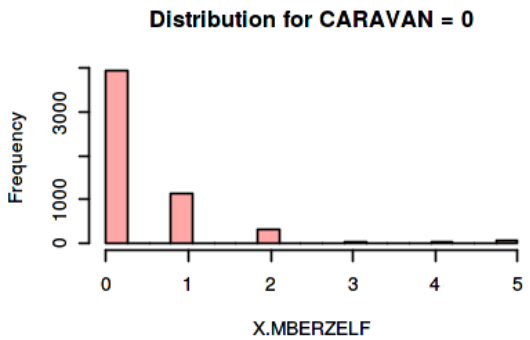
```
[1] "X.MBERZELF"
```

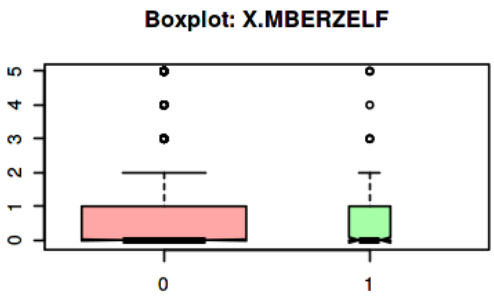
```
Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
1, 2, 0, 0, 0, :
“some notches went outside hinges ('box'): maybe set notch=FALSE”War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
“Chi-squared approximation may be incorrect”
```

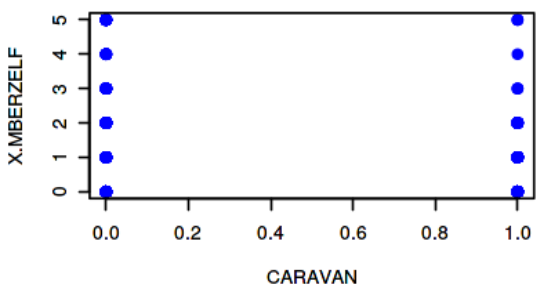
```
        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 4.4647, df = 5, p-value = 0.4846
```







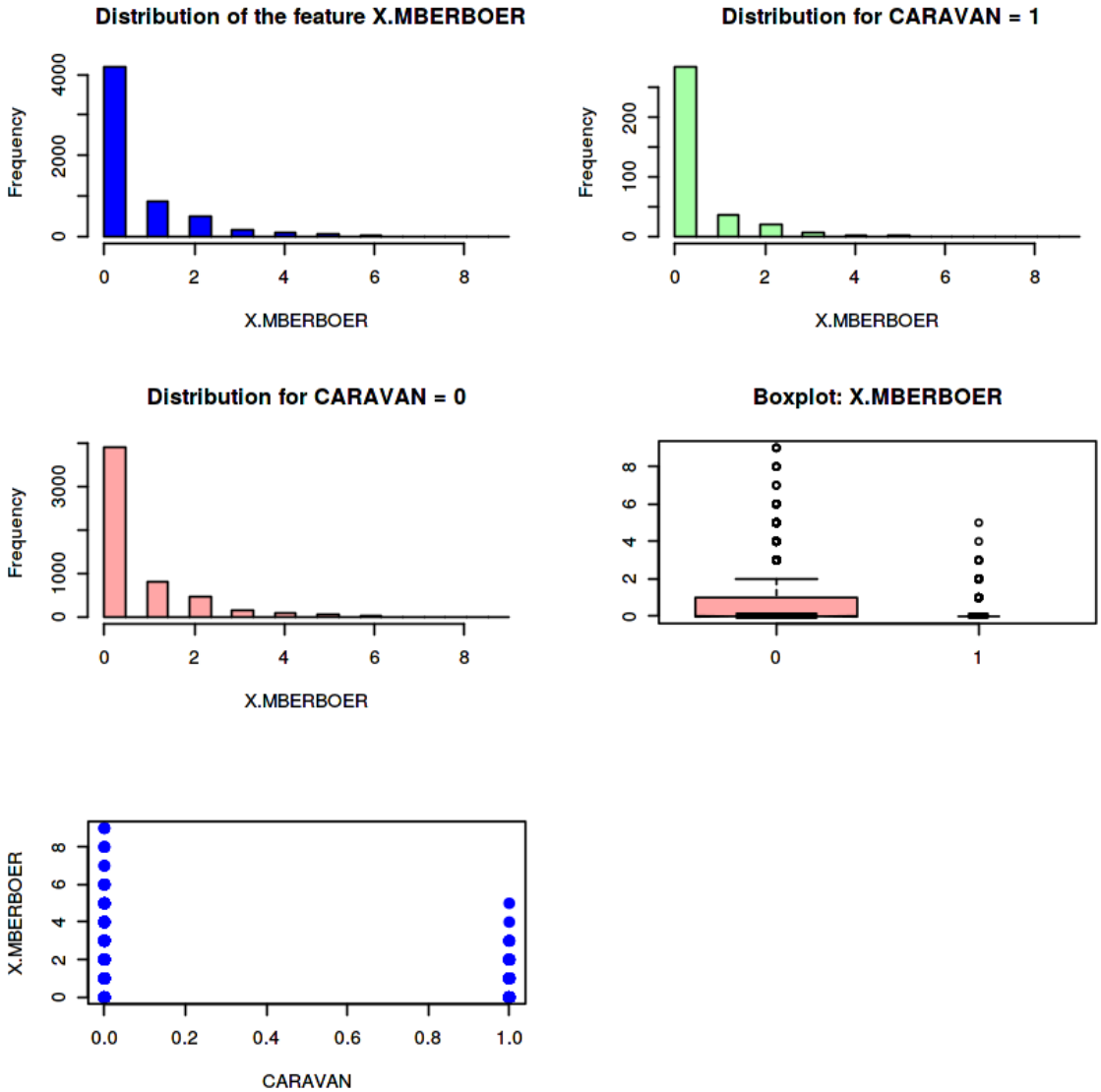




In [37]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[21]))
```

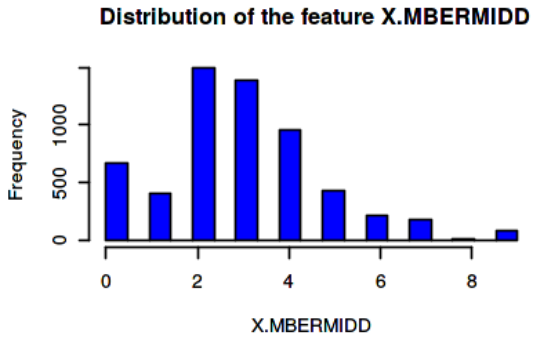
```
[1] "X.MBERBOER"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
1, 2, 0, 0, 0, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 20.172, df = 9, p-value = 0.01688
```

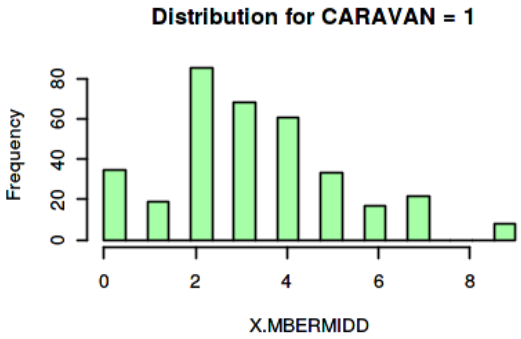


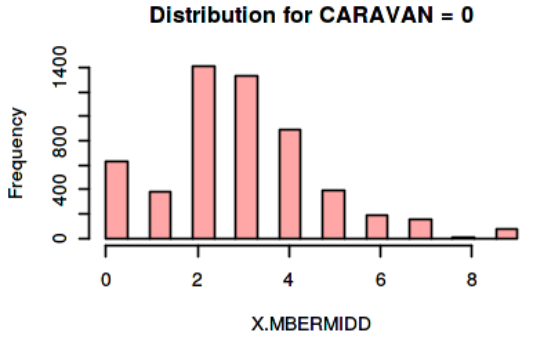
In [38]:

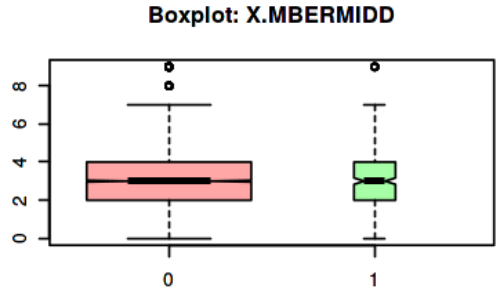
```
exploreFeature(train_data_cop,colnames(train_data_cop[22]))
```

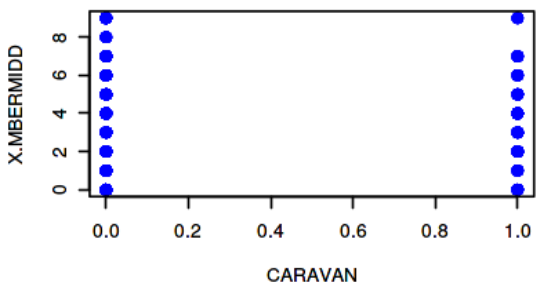
```
[1] "X.MBERMIDD"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 25.161, df = 9, p-value = 0.002798
```







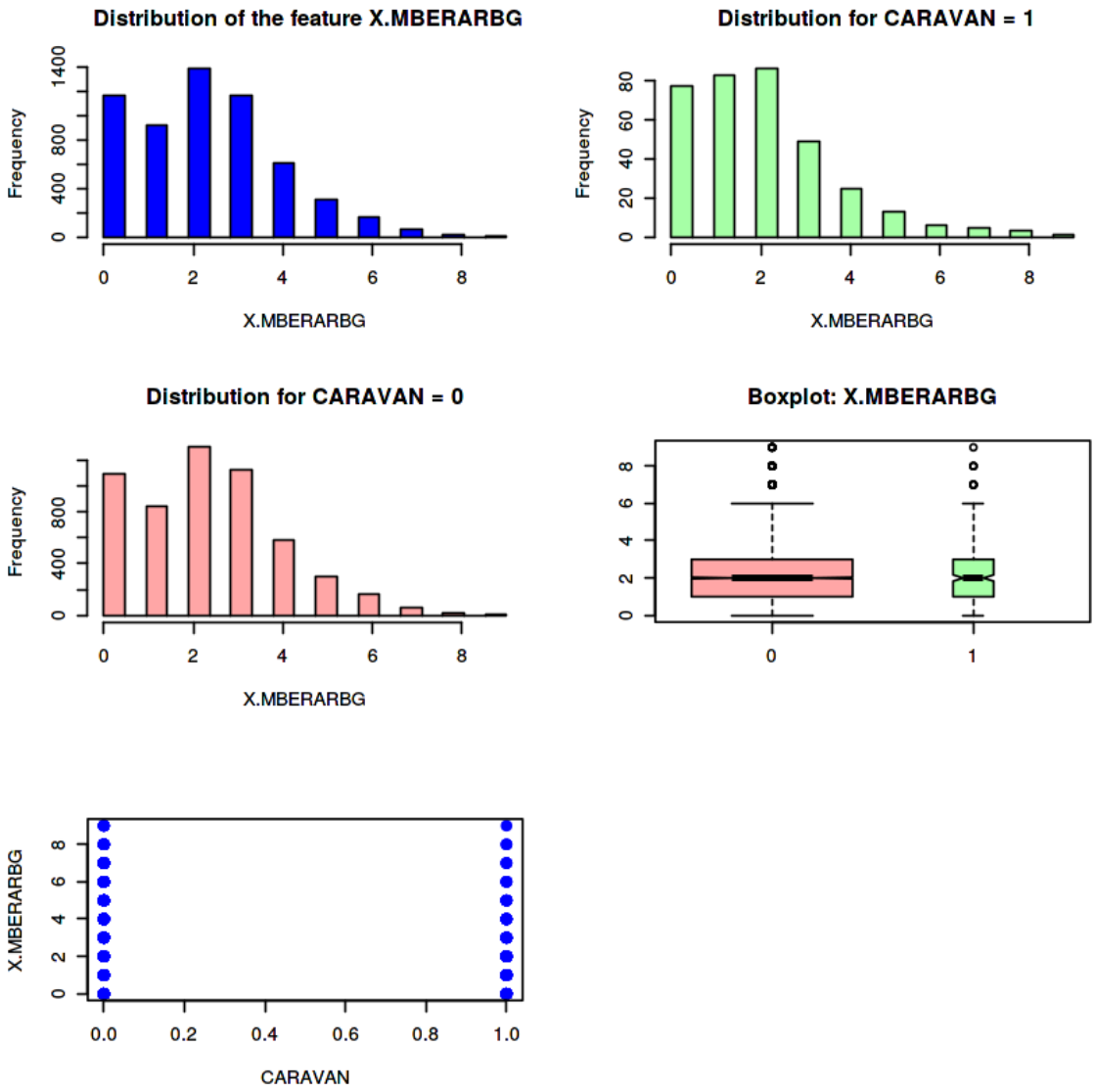




In [39]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[23]))
```

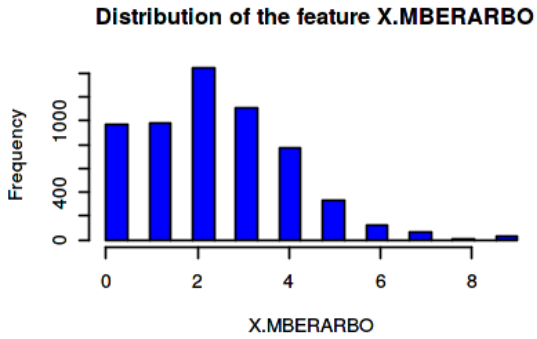
```
[1] "X.MBERARBG"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 32.1, df = 9, p-value = 0.0001914
```

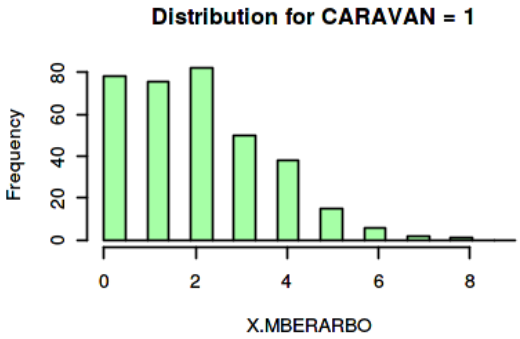


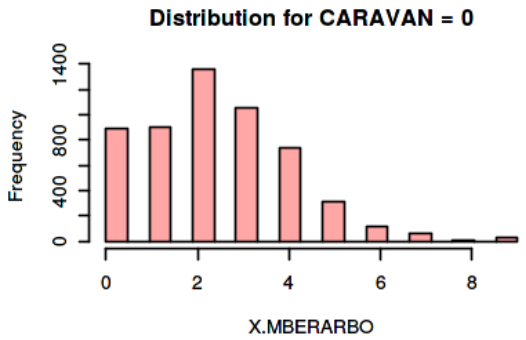
In [40]:

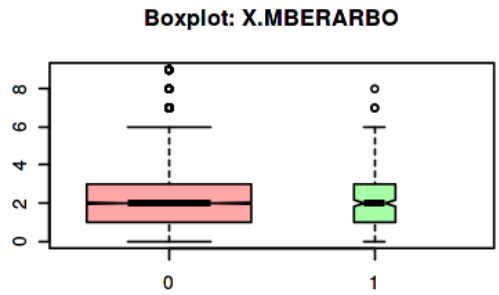
```
exploreFeature(train_data_cop,colnames(train_data_cop[24]))
```

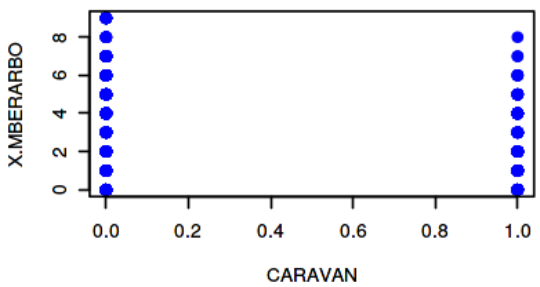
```
[1] "X.MBERARBO"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 23.521, df = 9, p-value = 0.005126
```







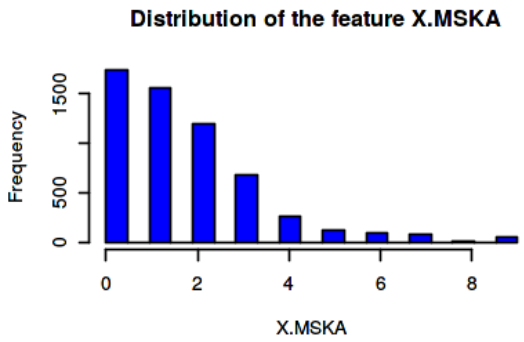


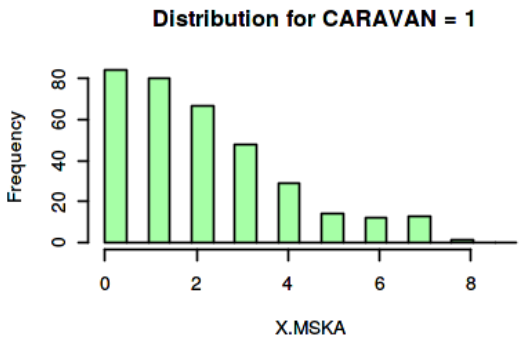


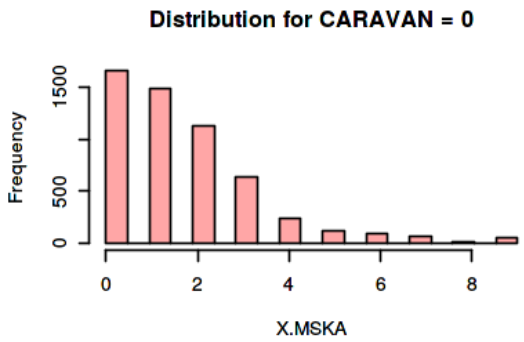
In [41]:

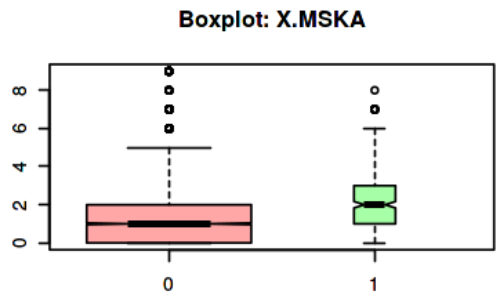
```
exploreFeature(train_data_cop,colnames(train_data_cop[25]))
```

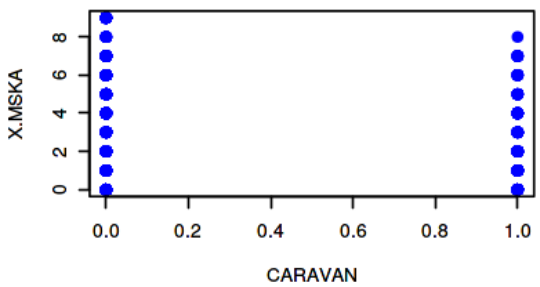
```
[1] "X.MSKA"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 52.136, df = 9, p-value = 4.261e-08
```







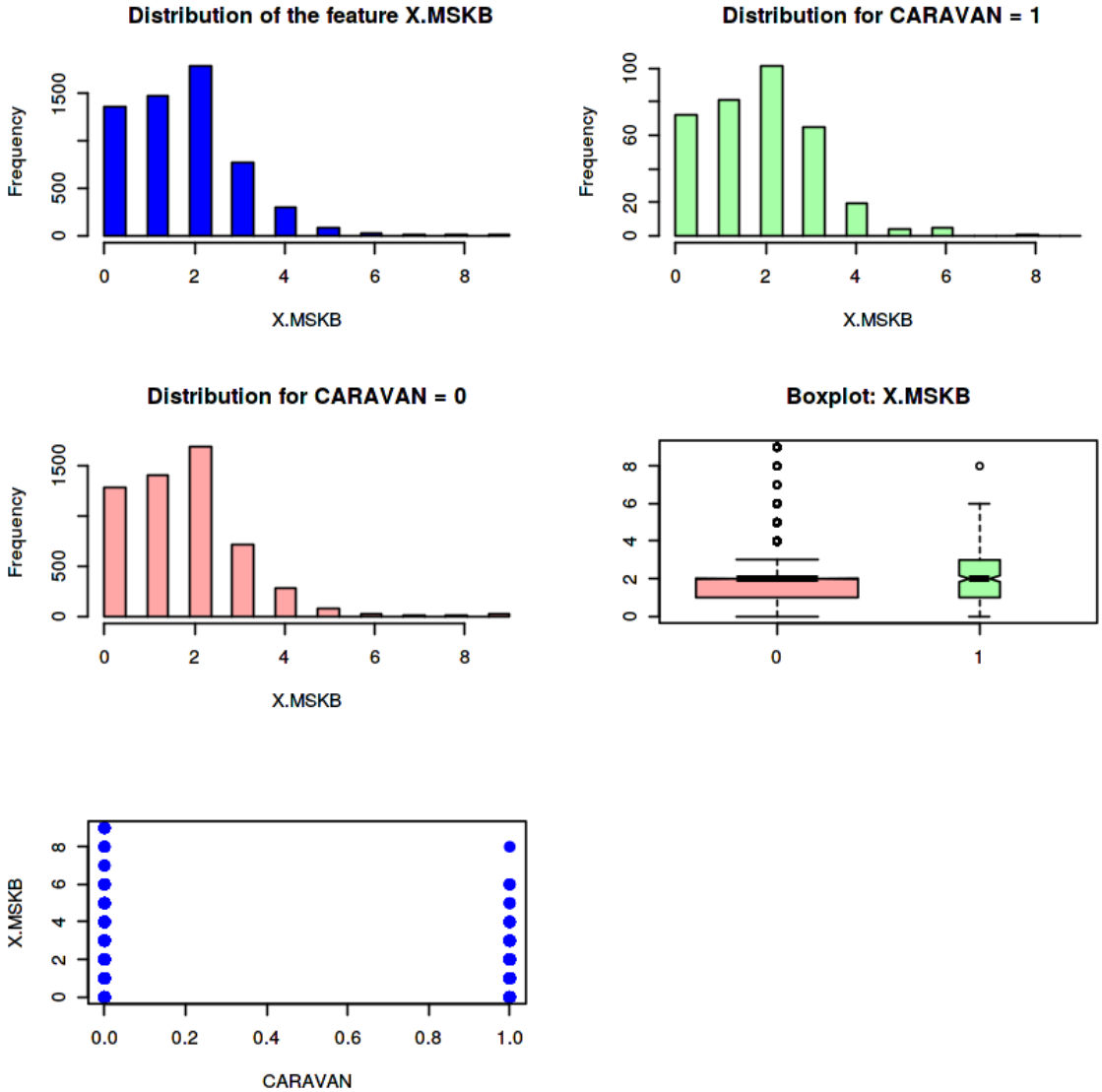




In [42]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[26]))
```

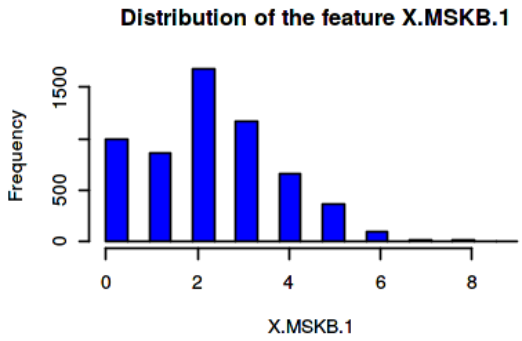
```
[1] "X.MSKB"

Warning message in bxp(structure(list(stats = structure(c(0, 1, 2,
2, 3, 0, 1, 2, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"


        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 20.959, df = 9, p-value = 0.01283
```

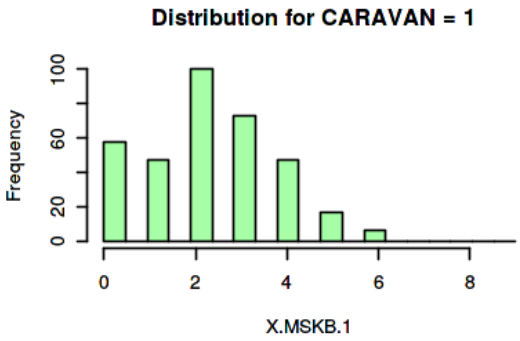


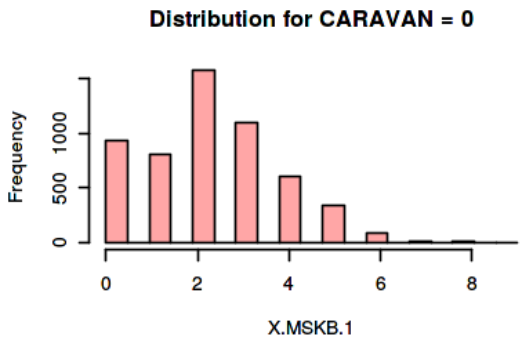
In [43]:

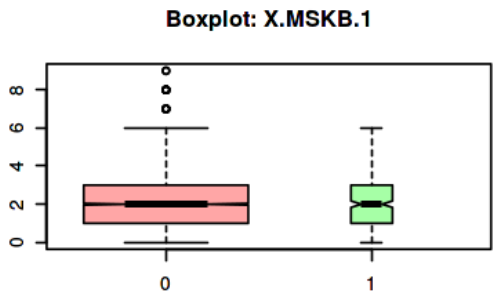
```
exploreFeature(train_data_cop,colnames(train_data_cop[27]))
```

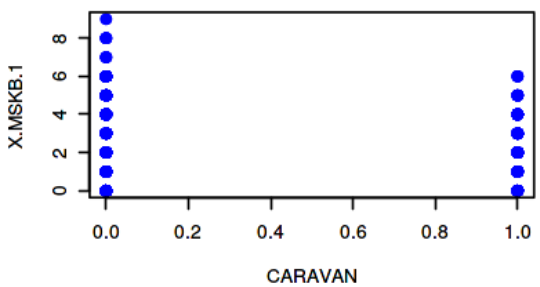
```
[1] "X.MSKB.1"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 4.2165, df = 9, p-value = 0.8966
```







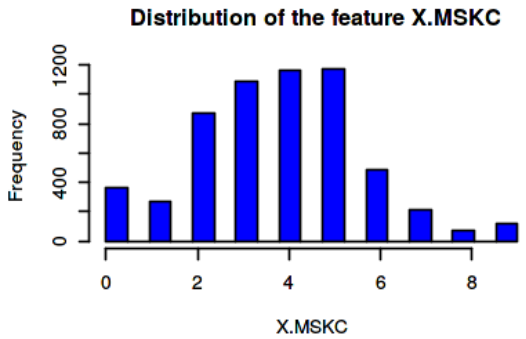


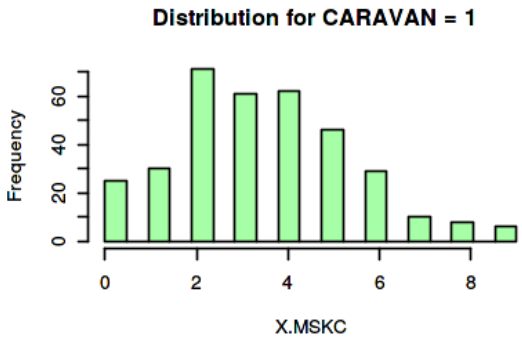


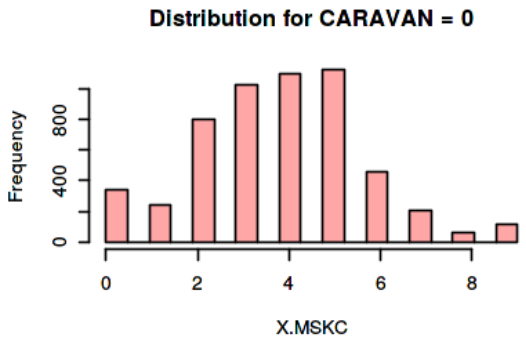
In [44]:

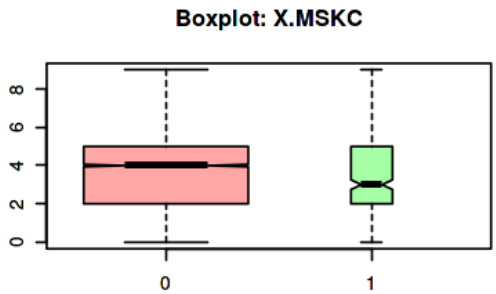
```
exploreFeature(train_data_cop,colnames(train_data_cop[28]))
```

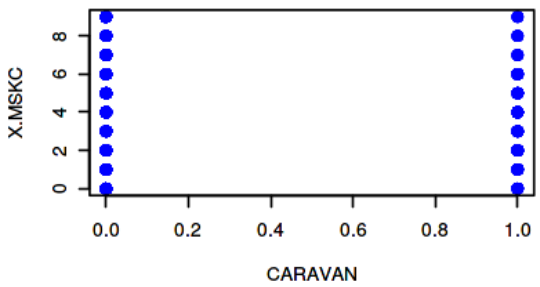
```
[1] "X.MSKC"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 34.529, df = 9, p-value = 7.213e-05
```







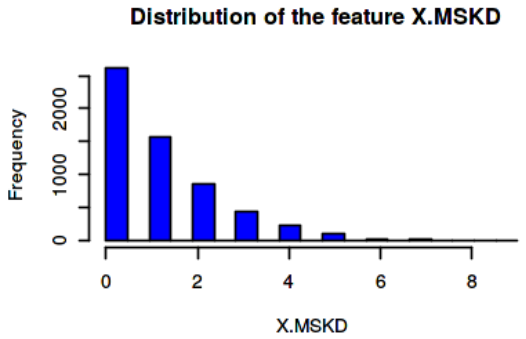


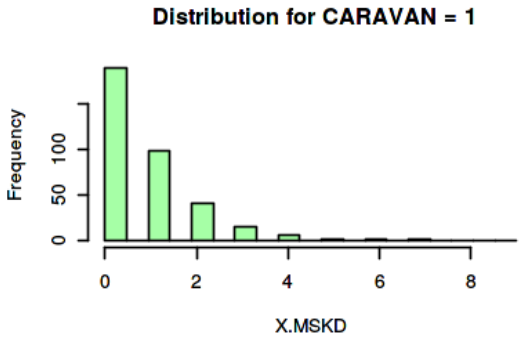


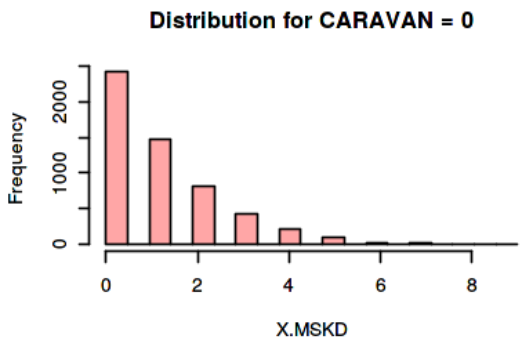
In [45]:

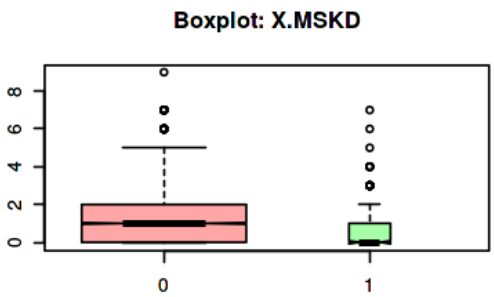
```
exploreFeature(train_data_cop,colnames(train_data_cop[29]))
```

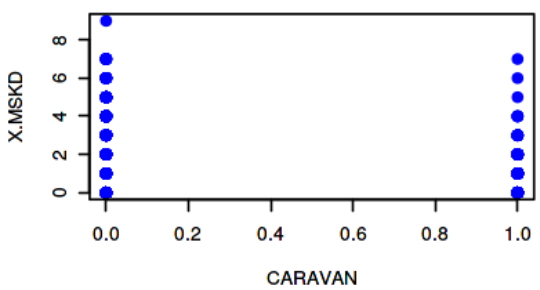
```
[1] "X.MSKD"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 1,
2, 5, 0, 0, 0, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"


        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 26.116, df = 8, p-value = 0.001004
```







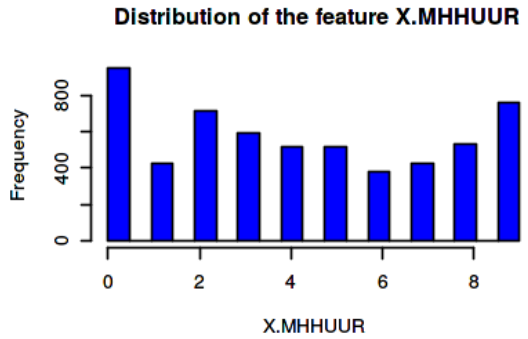


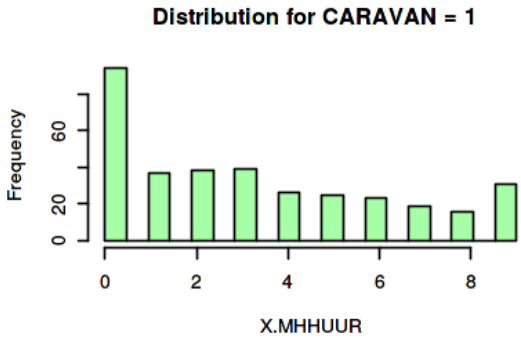


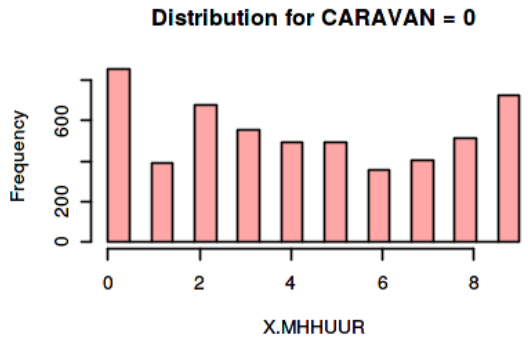
In [46]:

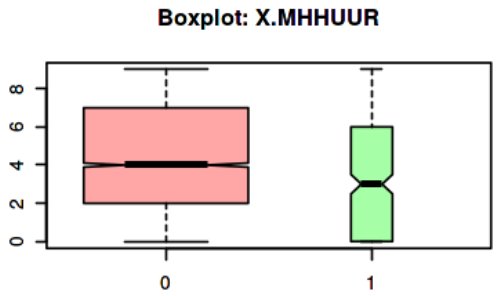
```
exploreFeature(train_data_cop,colnames(train_data_cop[30]))
```

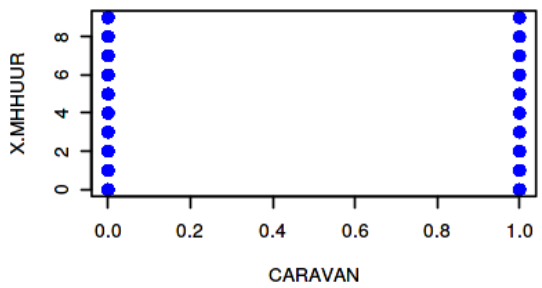
```
[1] "X.MHHUUR"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 49.446, df = 9, p-value = 1.369e-07
```







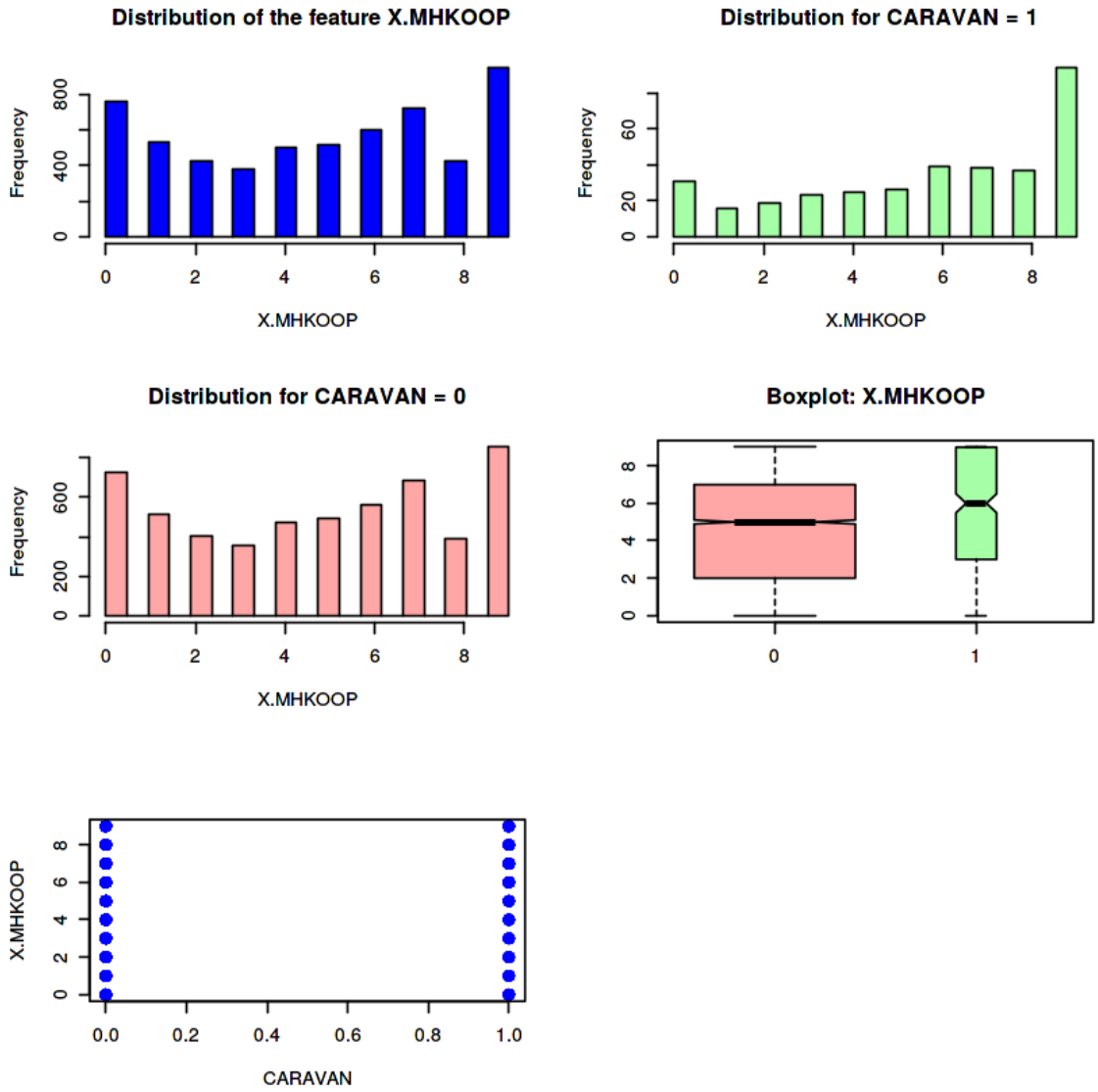




In [47]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[31]))
```

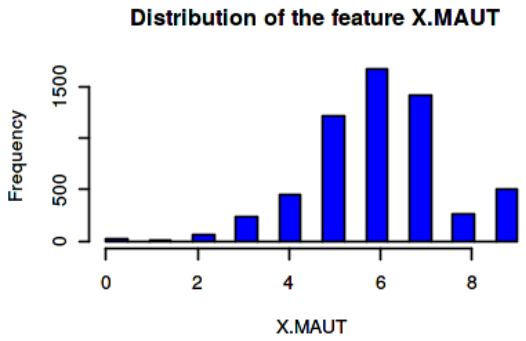
```
[1] "X.MHKOOP"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 48.99, df = 9, p-value = 1.667e-07
```

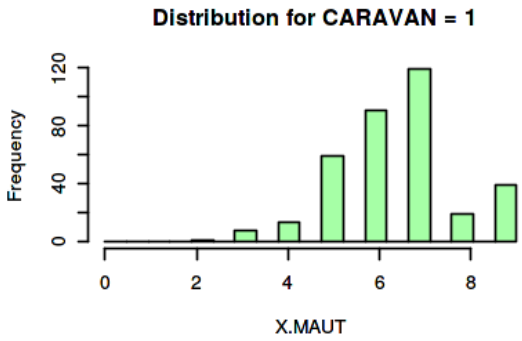


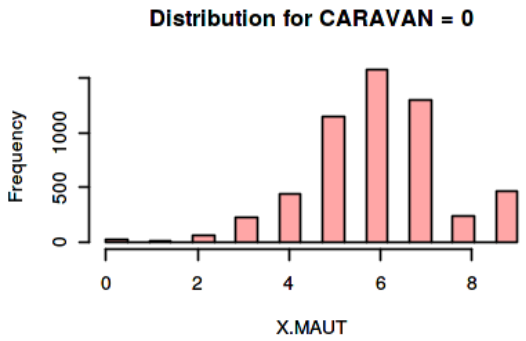
In [48]:

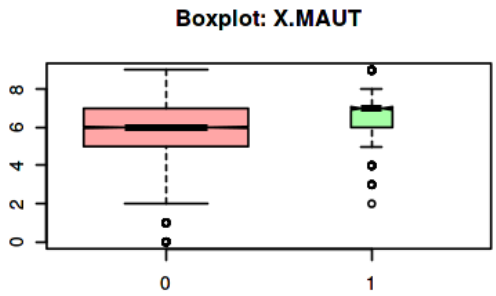
```
exploreFeature(train_data_cop,colnames(train_data_cop[32]))
```

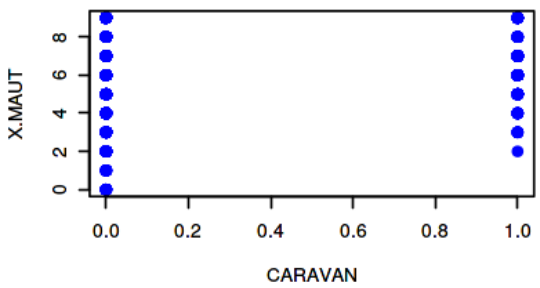
```
[1] "X.MAUT"

Warning message in bxp(structure(list(stats = structure(c(2, 5, 6,
7, 9, 5, 6, 7, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"


        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 36.99, df = 9, p-value = 2.642e-05
```







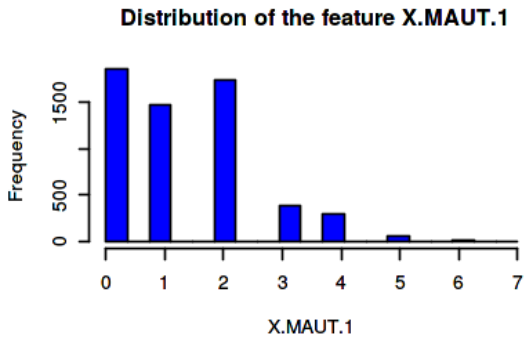


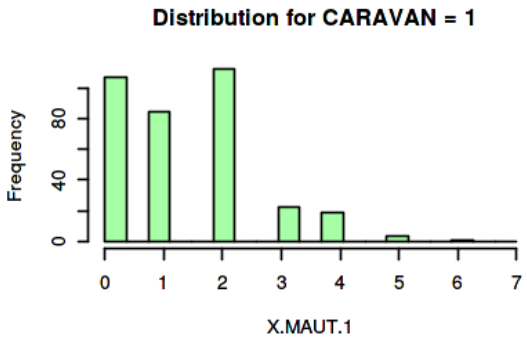


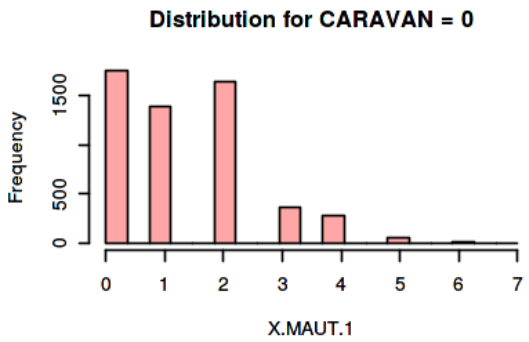
In [49]:

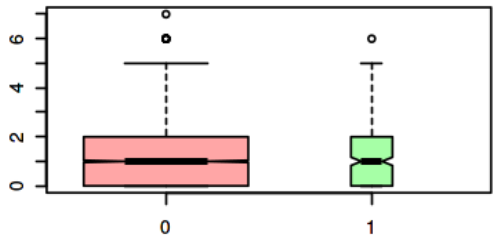
```
exploreFeature(train_data_cop,colnames(train_data_cop[33]))
```

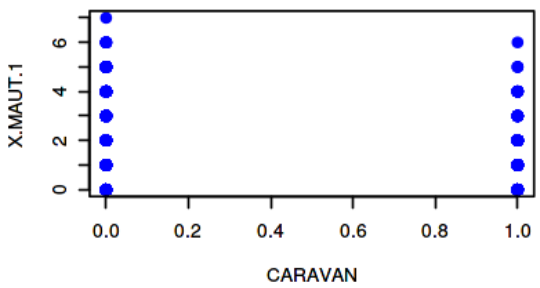
```
[1] "X.MAUT.1"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 1.5168, df = 7, p-value = 0.9817
```







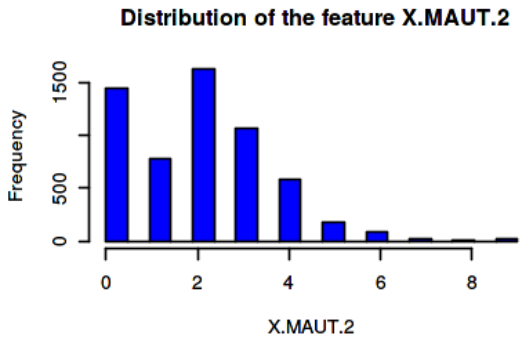


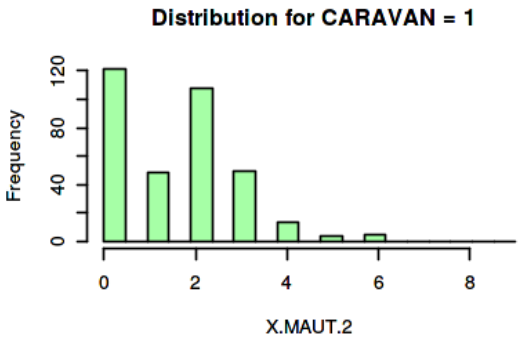


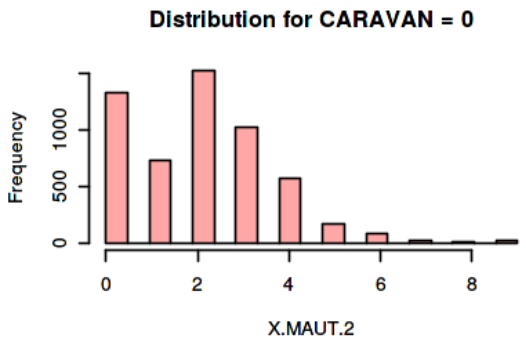
In [50]:

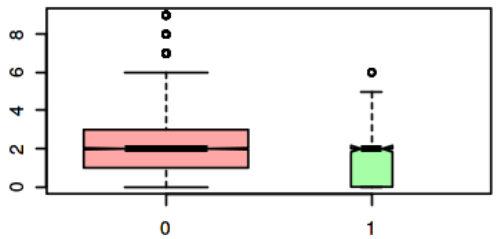
```
exploreFeature(train_data_cop,colnames(train_data_cop[34]))
```

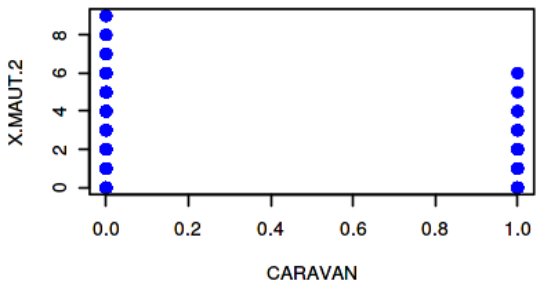
```
[1] "X.MAUT.2"

Warning message in bxp(structure(list(stats = structure(c(0, 1, 2,
3, 6, 0, 0, 2, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 41.924, df = 9, p-value = 3.393e-06
```







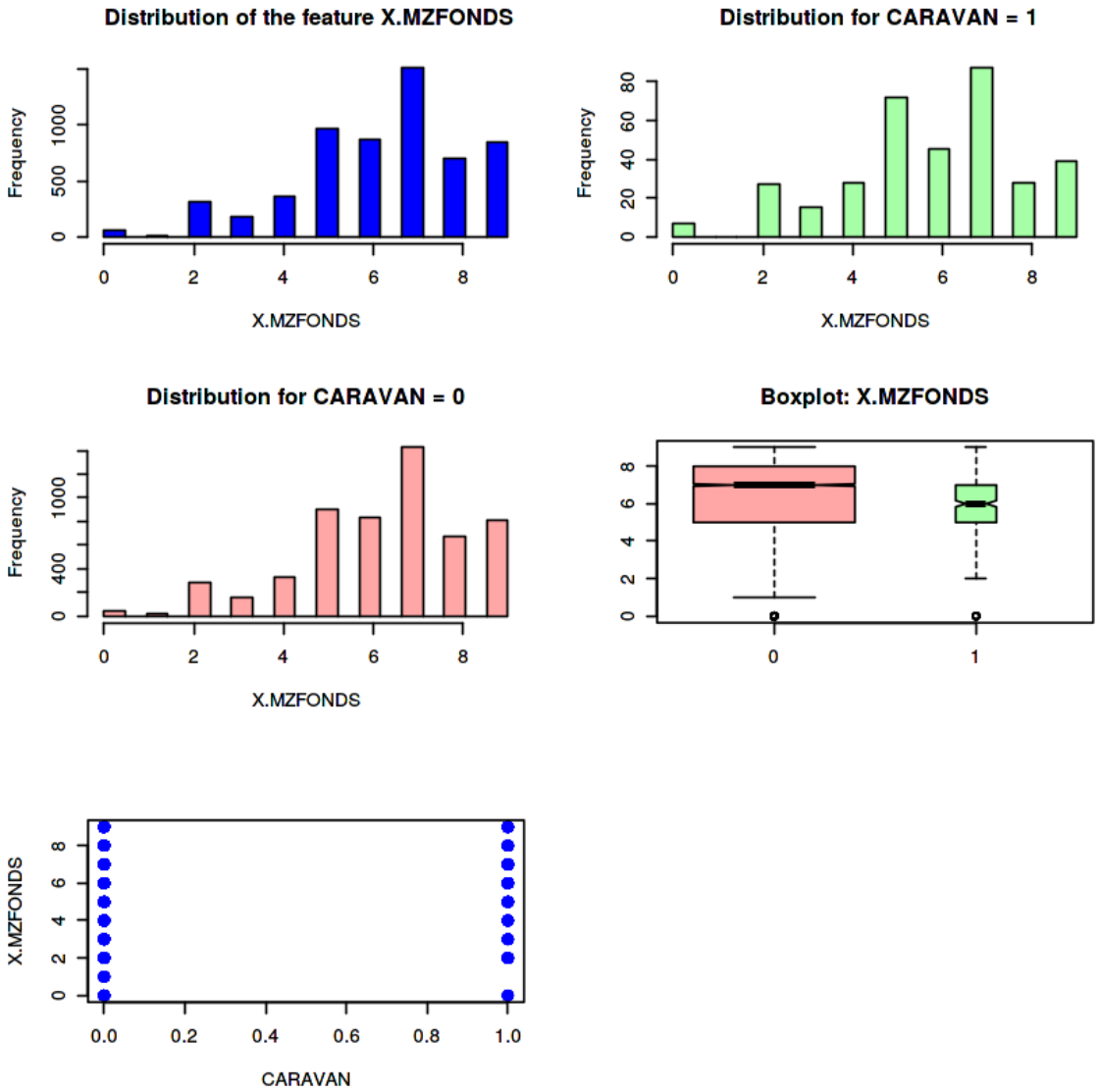




In [51]:

```
exploreFeature(train_data_cop,colnames(train_data_cop[35]))
```

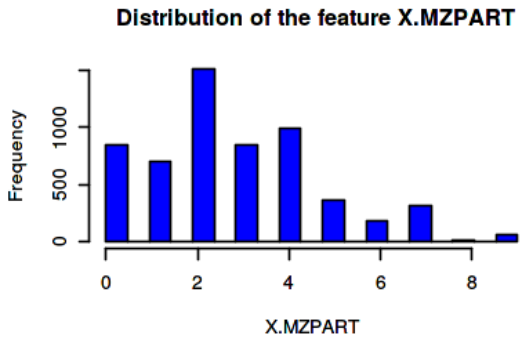
```
[1] "X.MZFONDS"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 26.413, df = 9, p-value = 0.001748
```

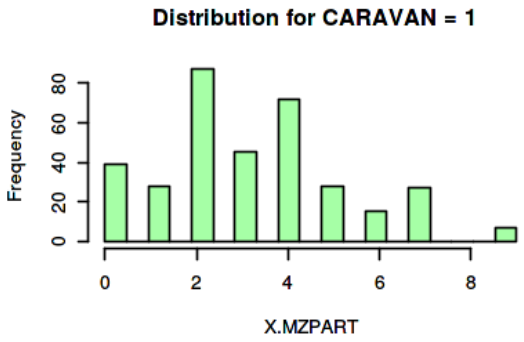


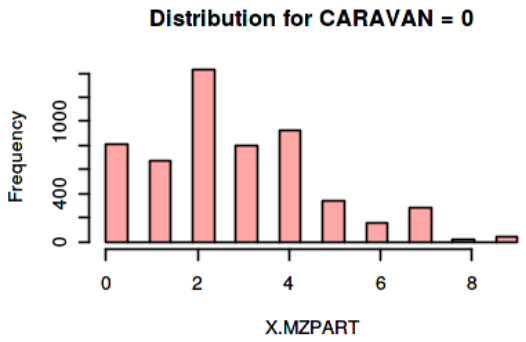
In [52]:

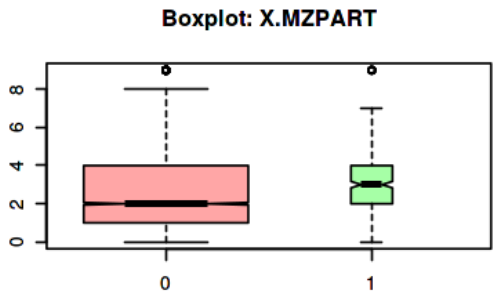
```
exploreFeature(train_data_cop,colnames(train_data_cop[36]))
```

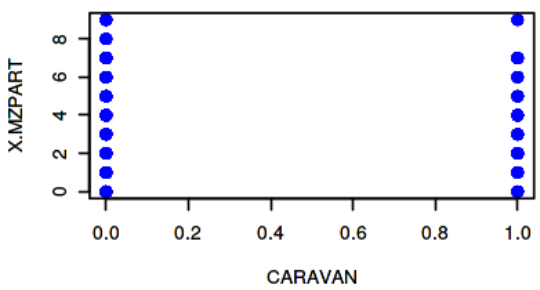
```
[1] "X.MZPART"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 25.077, df = 9, p-value = 0.002887
```







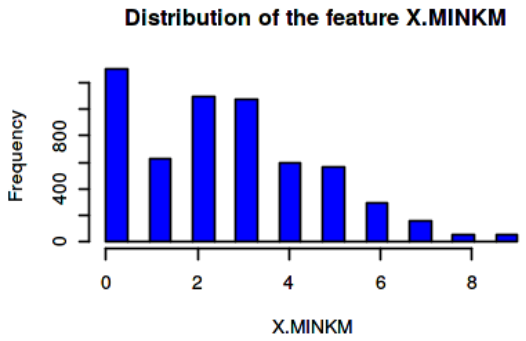


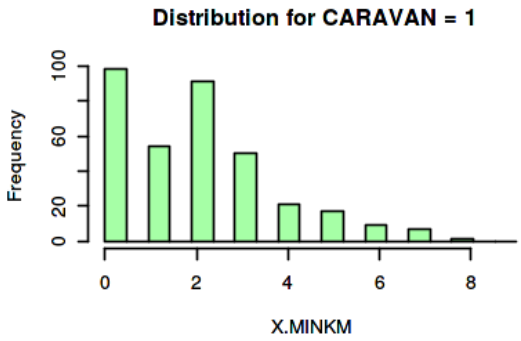


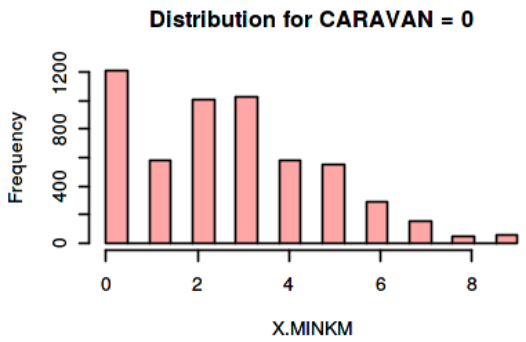
In [53]:

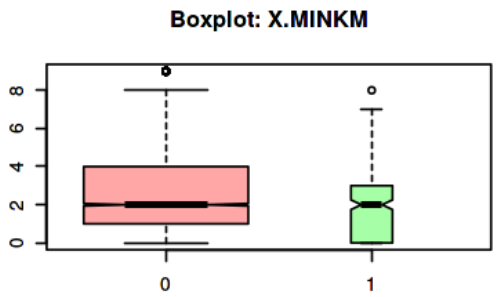
```
exploreFeature(train_data_cop,colnames(train_data_cop[37]))
```

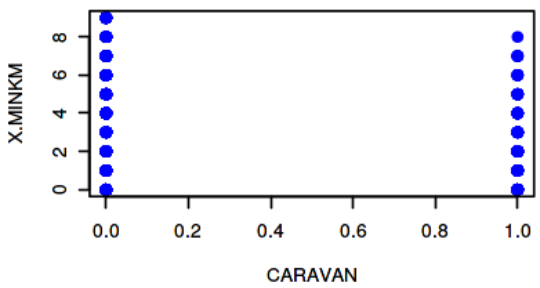
```
[1] "X.MINKM"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 52.23, df = 9, p-value = 4.089e-08
```







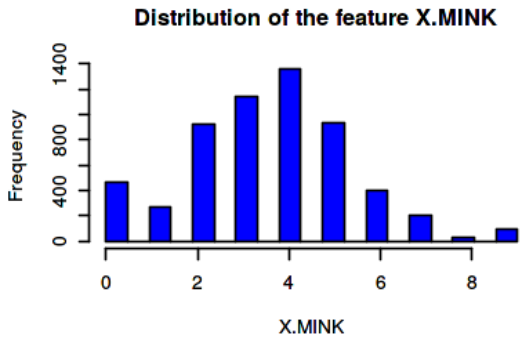


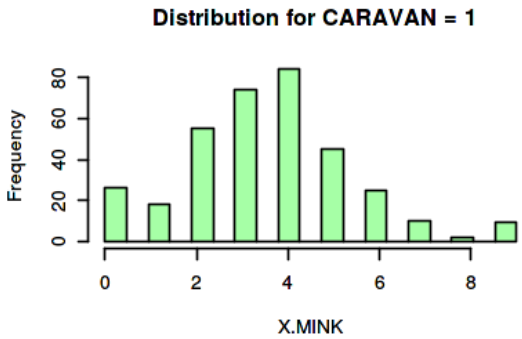


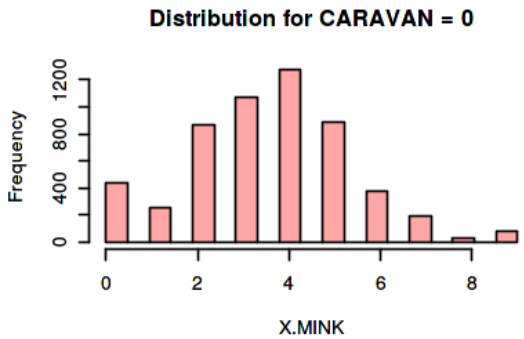
In [54]:

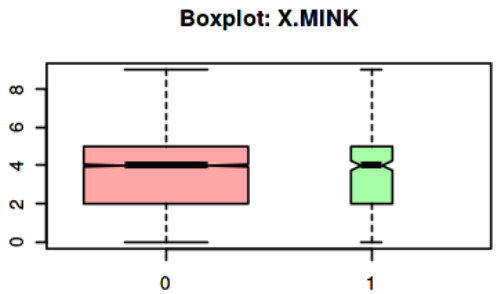
```
exploreFeature(train_data_cop,colnames(train_data_cop[38]))
```

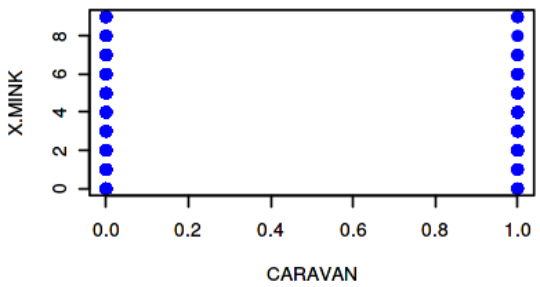
```
[1] "X.MINK"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 6.1842, df = 9, p-value = 0.7214
```







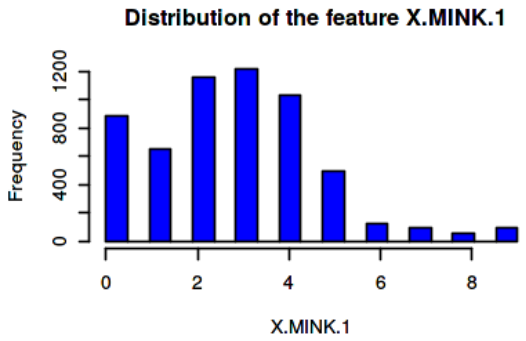


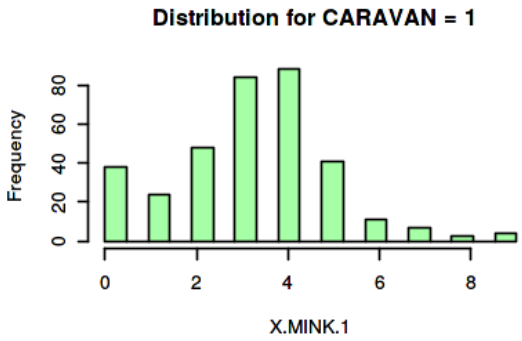


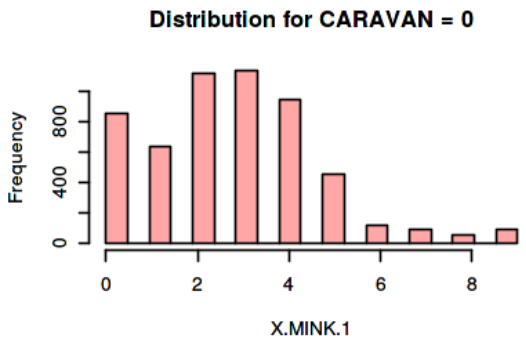
In [55]:

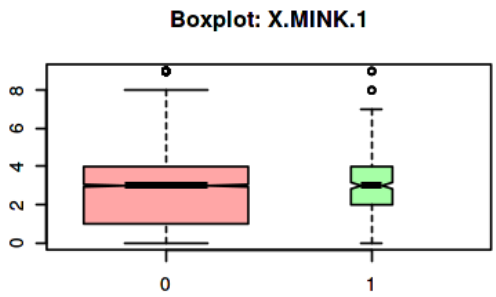
```
exploreFeature(train_data_cop,colnames(train_data_cop[39]))
```

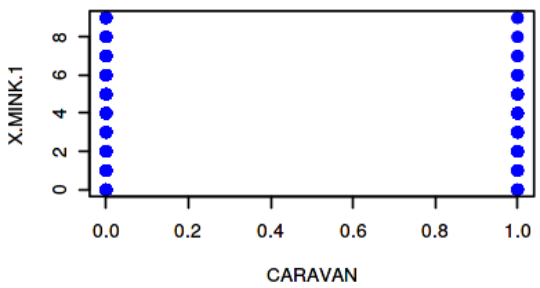
```
[1] "X.MINK.1"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 38.91, df = 9, p-value = 1.196e-05
```







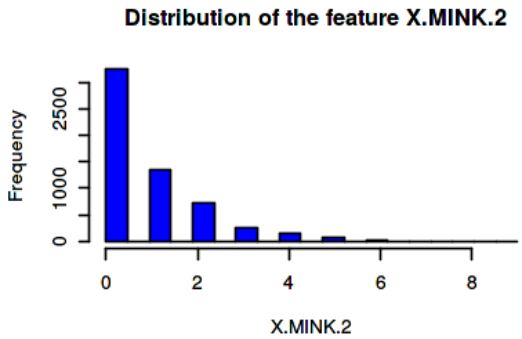


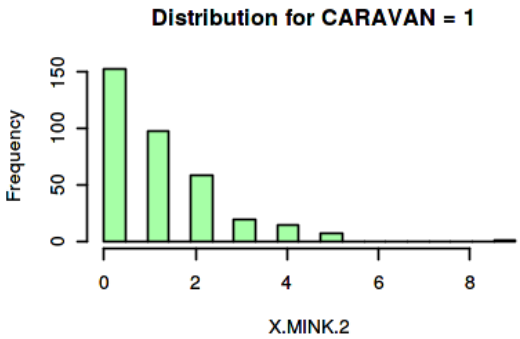


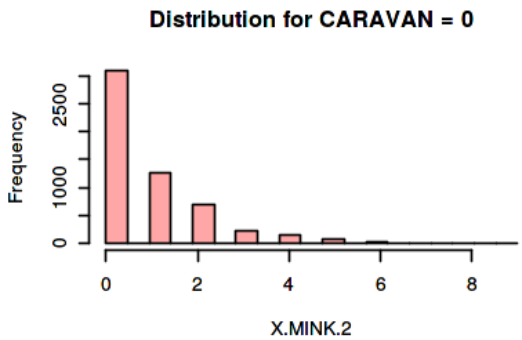
In [56]:

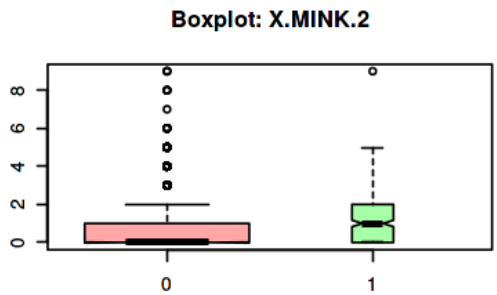
```
exploreFeature(train_data_cop,colnames(train_data_cop[40]))
```

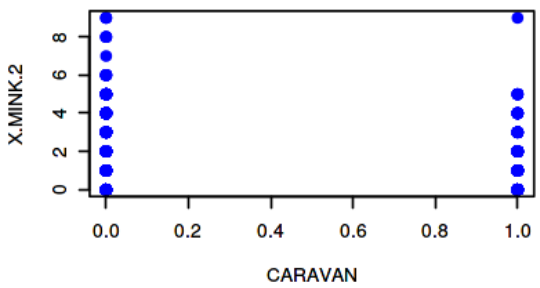
```
[1] "X.MINK.2"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
1, 2, 0, 0, 1, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 27.609, df = 9, p-value = 0.001108
```







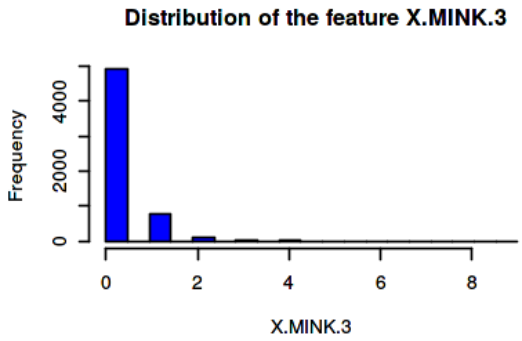


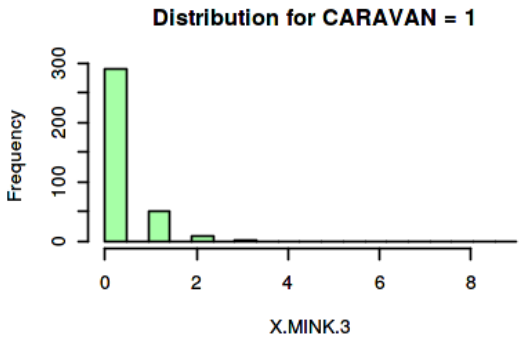


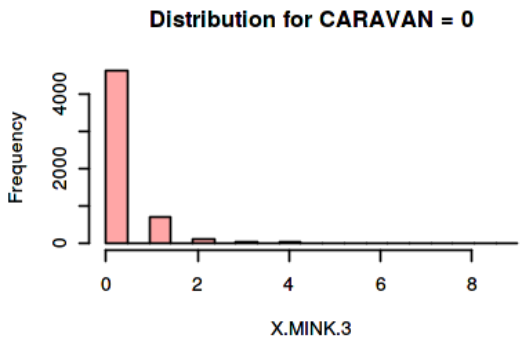
In [57]:

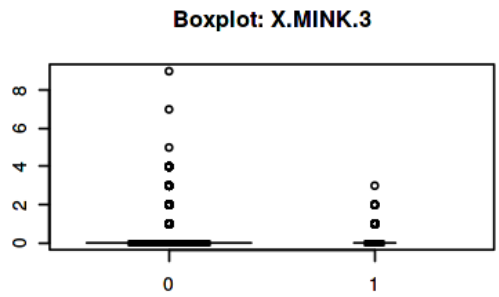
```
exploreFeature(train_data_cop,colnames(train_data_cop[41]))
```

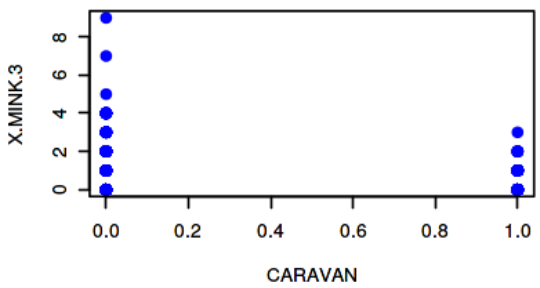
```
[1] "X.MINK.3"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 3.8254, df = 7, p-value = 0.7997
```







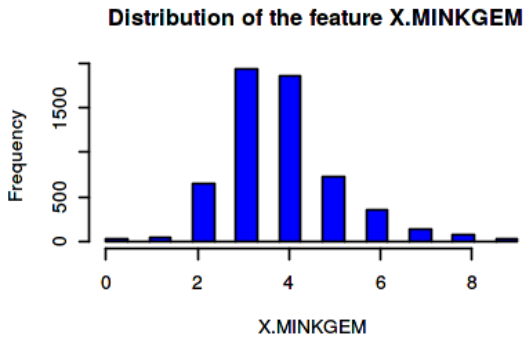


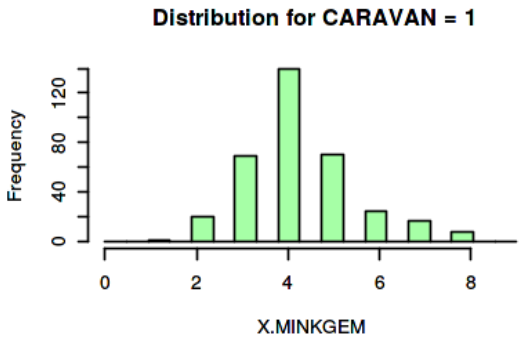


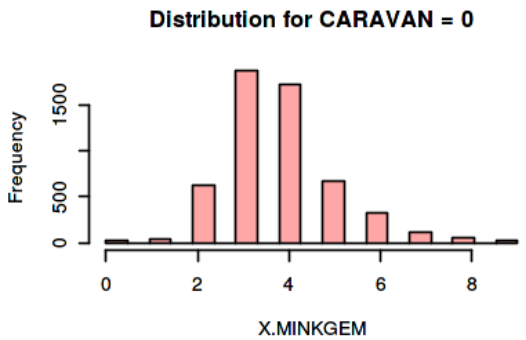
In [58]:

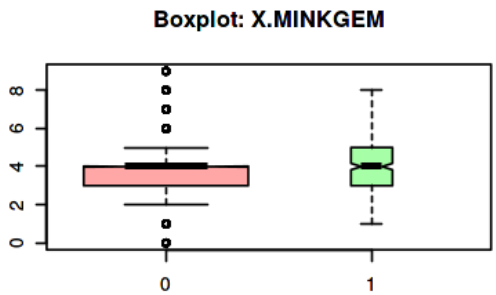
```
exploreFeature(train_data_cop,colnames(train_data_cop[42]))
```

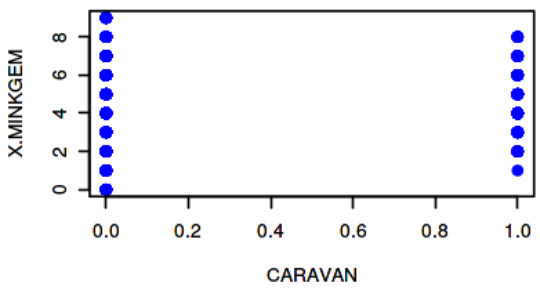
```
[1] "X.MINKGEM"

Warning message in bxp(structure(list(stats = structure(c(2, 3, 4,
4, 5, 1, 3, 4, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 73.791, df = 9, p-value = 2.737e-12
```







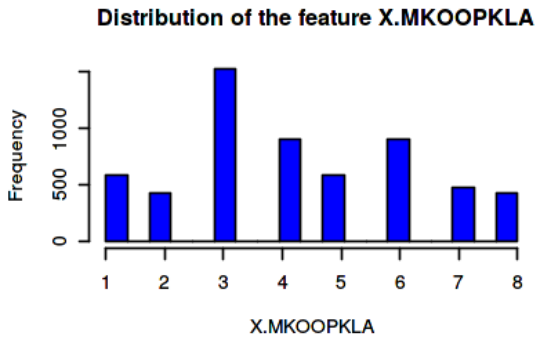


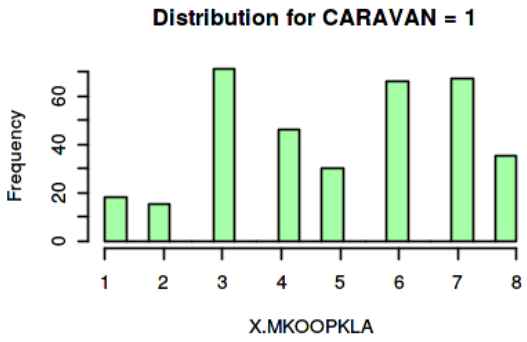


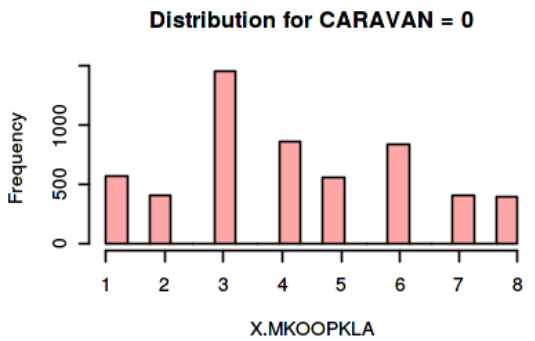
In [59]:

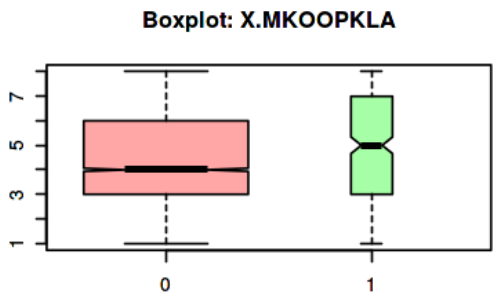
```
exploreFeature(train_data_cop,colnames(train_data_cop[43]))
```

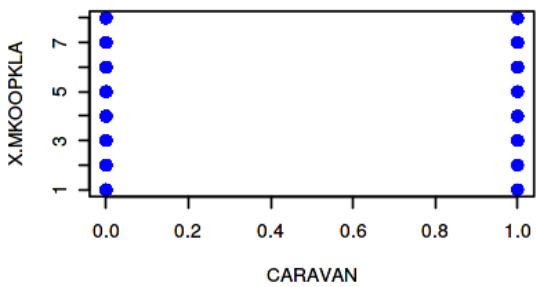
```
[1] "X.MKOOPKLA"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 82.888, df = 7, p-value = 3.545e-15
```







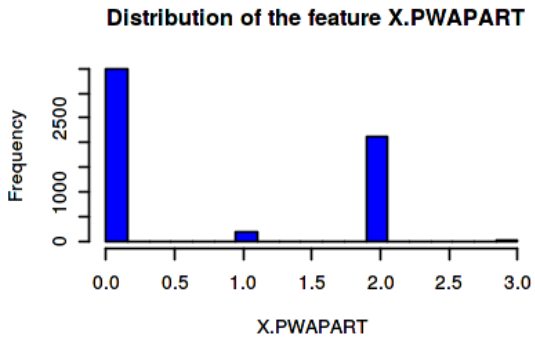


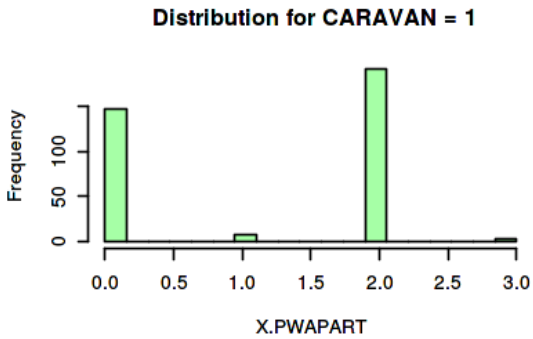


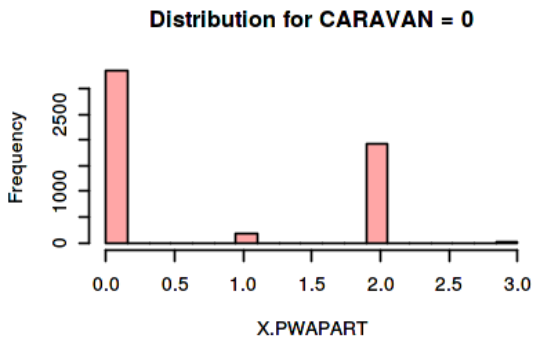
In [60]:

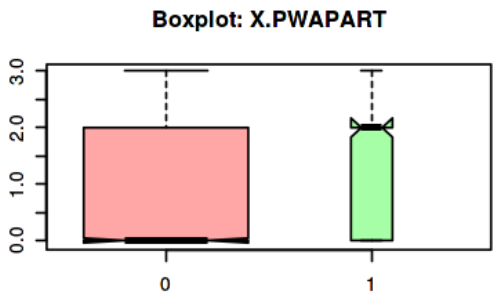
```
exploreFeature(train_data_cop,colnames(train_data_cop[44]))
```

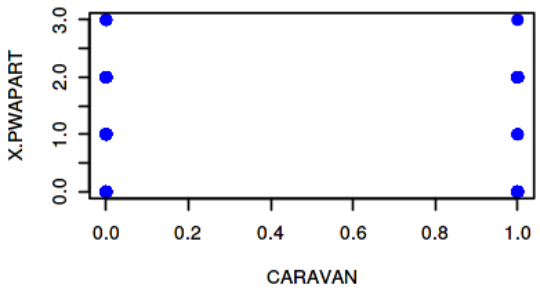
```
[1] "X.PWAPART"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
2, 3, 0, 0, 2, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 57.476, df = 3, p-value = 2.034e-12
```







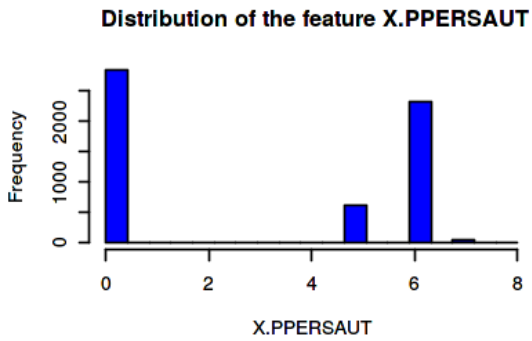


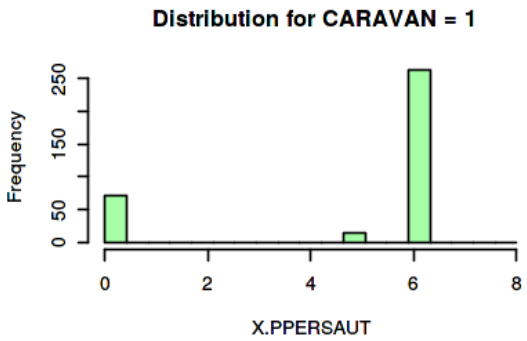


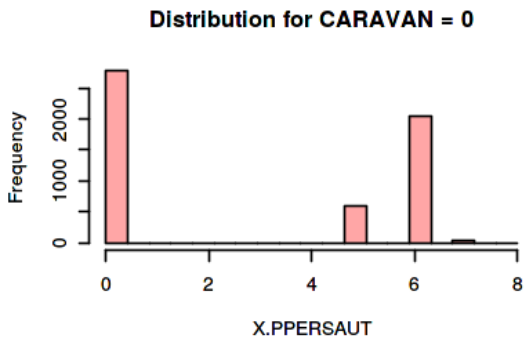
In [61]:

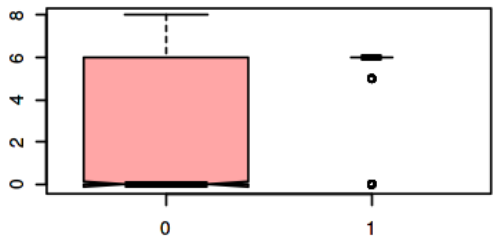
```
exploreFeature(train_data_cop,colnames(train_data_cop[45]))
```

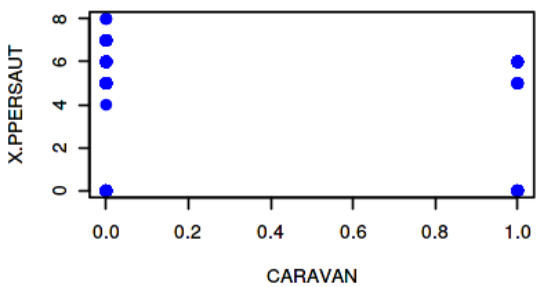
```
[1] "X.PPERSAUT"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
6, 8, 6, 6, 6, :
“some notches went outside hinges ('box'): maybe set notch=FALSE”War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
“Chi-squared approximation may be incorrect”

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 194.69, df = 5, p-value < 2.2e-16
```







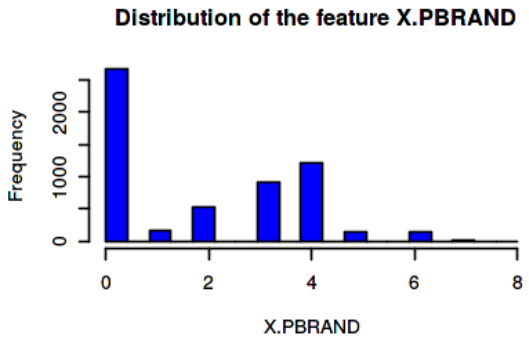


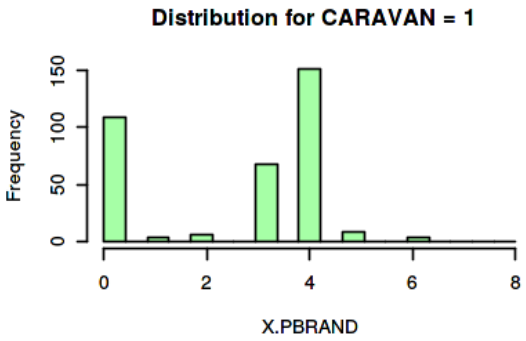


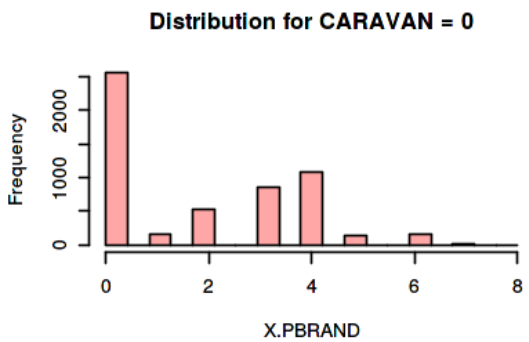
In [62]:

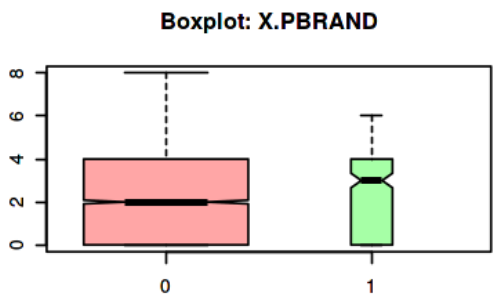
```
exploreFeature(train_data_cop,colnames(train_data_cop[46]))
```

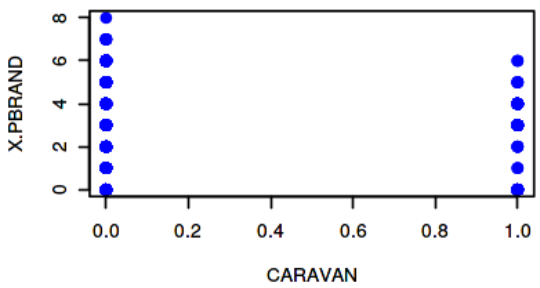
```
[1] "X.PBRAND"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 140.39, df = 8, p-value < 2.2e-16
```







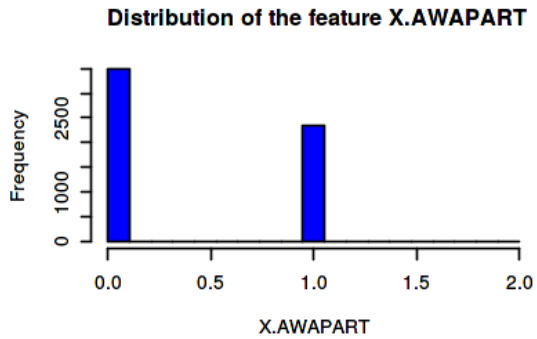


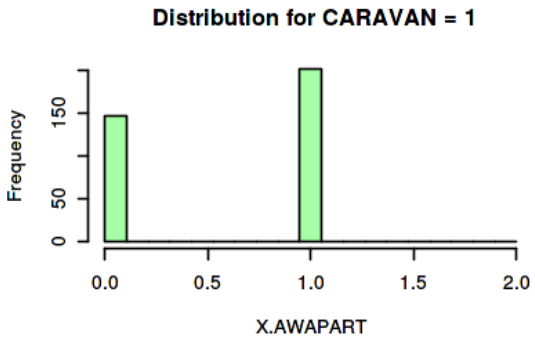


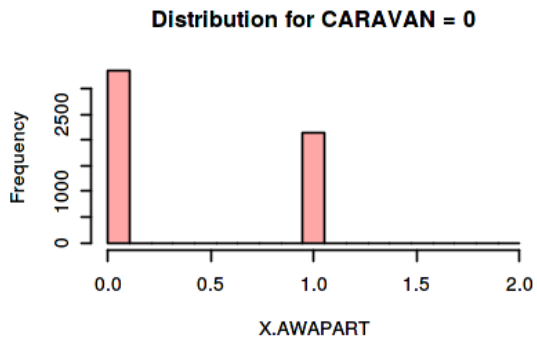
In [63]:

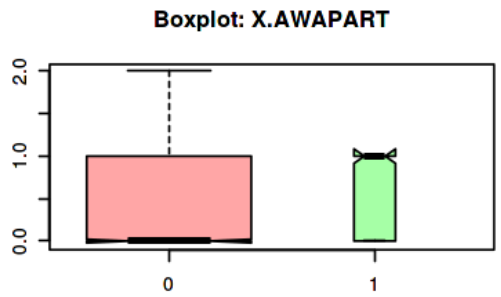
```
exploreFeature(train_data_cop,colnames(train_data_cop[47]))
```

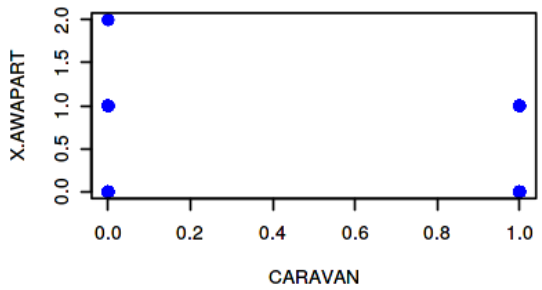
```
[1] "X.AWAPART"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
1, 2, 0, 0, 1, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 48.302, df = 2, p-value = 3.246e-11
```







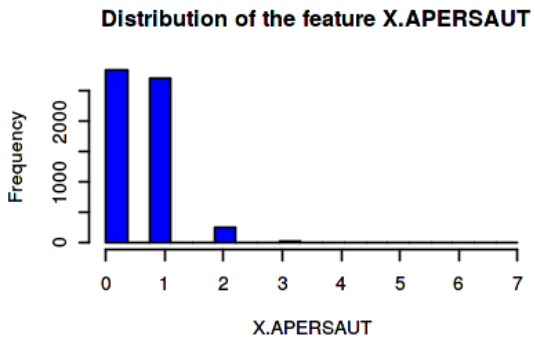


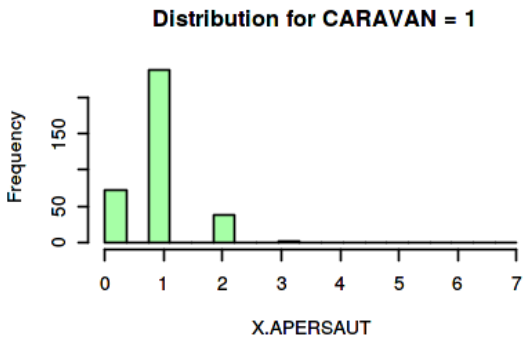


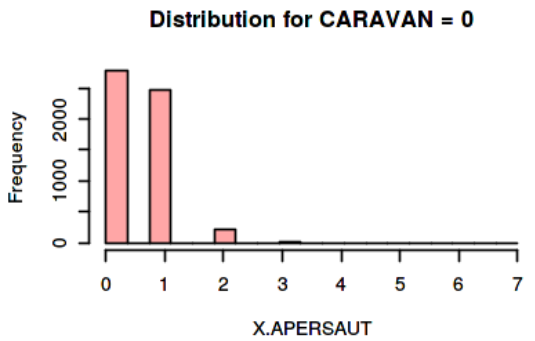
In [64]:

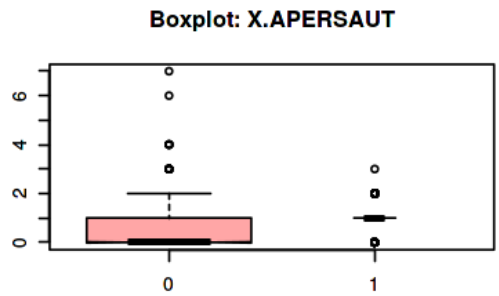
```
exploreFeature(train_data_cop,colnames(train_data_cop[48]))
```

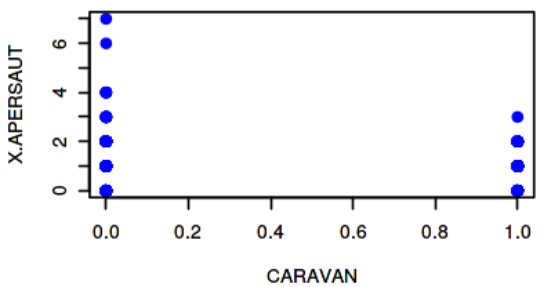
```
[1] "X.APERSAUT"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 0,
1, 2, 1, 1, 1, :
“some notches went outside hinges ('box'): maybe set notch=FALSE”War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
“Chi-squared approximation may be incorrect”

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 136.75, df = 6, p-value < 2.2e-16
```







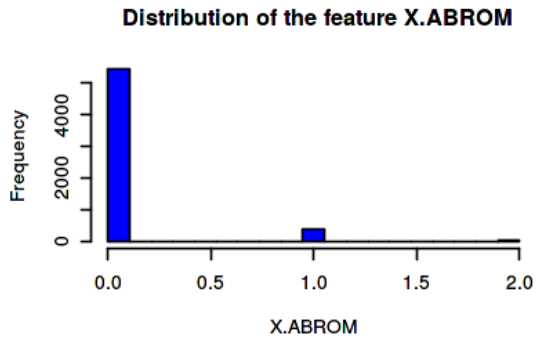


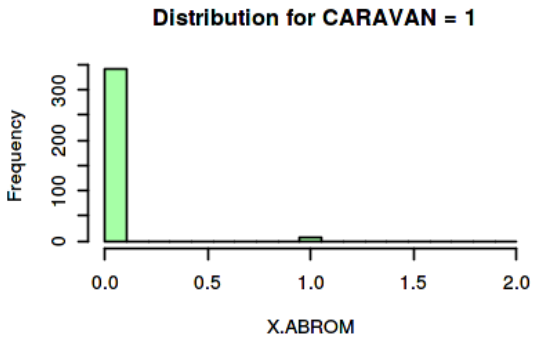


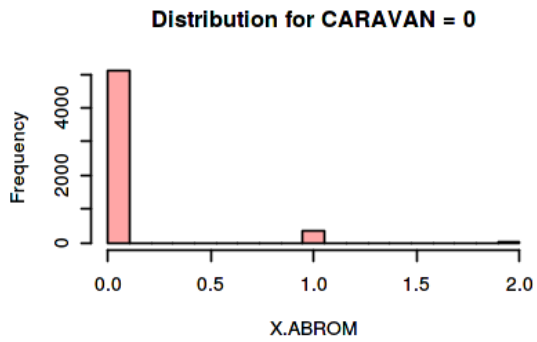
In [65]:

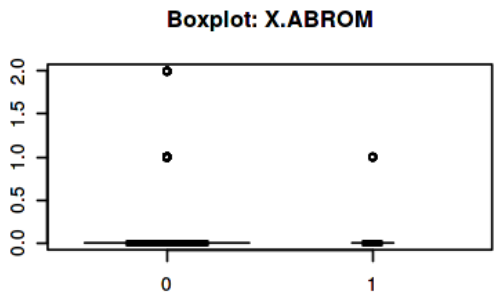
```
exploreFeature(train_data_cop,colnames(train_data_cop[49]))
```

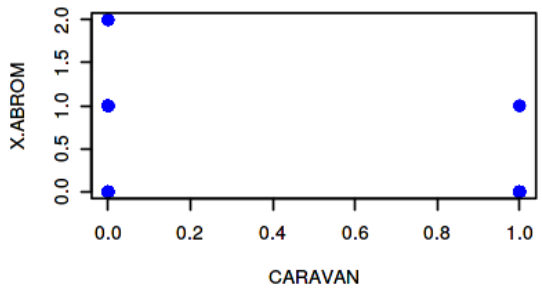
```
[1] "X.ABROM"

Warning message in chisq.test(train_data[, feature], train_data_cop
$X.CARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 11.944, df = 2, p-value = 0.002549
```







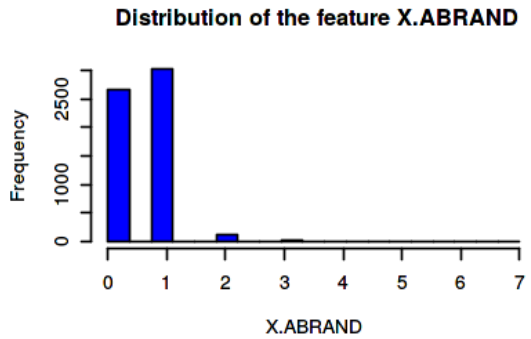


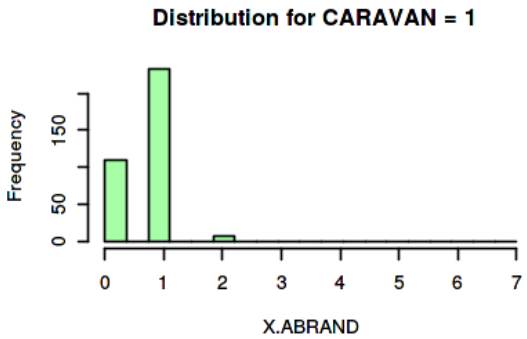


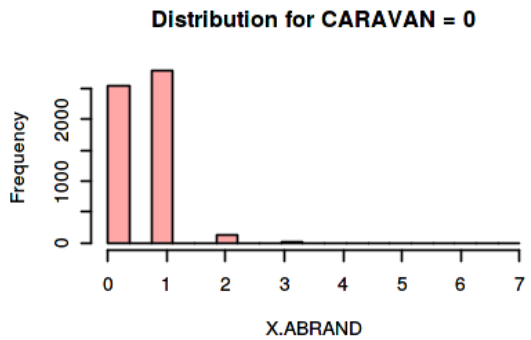
In [66]:

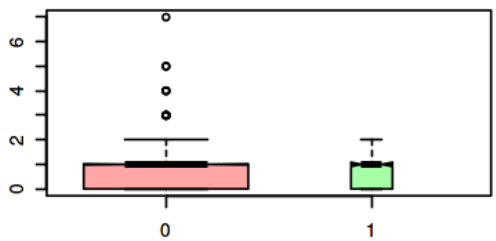
```
exploreFeature(train_data_cop,colnames(train_data_cop[50]))
```

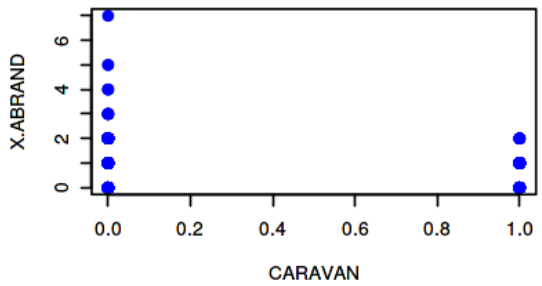
```
[1] "X.ABRAND"

Warning message in bxp(structure(list(stats = structure(c(0, 0, 1,
1, 2, 0, 0, 1, :
"some notches went outside hinges ('box'): maybe set notch=FALSE"War
ning message in chisq.test(train_data[, feature], train_data_cop$X.C
ARAVAN):
"Chi-squared approximation may be incorrect"

        Pearson's Chi-squared test

data:  train_data[, feature] and train_data_cop$X.CARAVAN
X-squared = 33.532, df = 6, p-value = 8.282e-06
```











After in depth analysis for each of the feature we have decided 26 features that seem to be largely indicative of interpreting the response variable. Following was the criteria for choosing the features:

- ***The overall distribution** : The distribution of the feature which shows gaussian-like distribution. More evenly distributed the feature elements are, the more variance it provides and more is the capability to interpret the reponse variable.*
- ***A sense of demarcation between the distribution plots** with respect to CARAVAN = 0 and CARAVAN = 1. Although, considering the scale of the frequencies may differ among the sub categories, However, the presence of similar pattern in the distribution indicates the proportionality in the distribution and nullification of the presence of prominent features. Imbalance in the distribution of the histograms shows a certain sense of entropy in the feature. And this entropy helps in identifying the likelihood of a customer buying a Caravan Policy.*
- ***Clear divisions of notches in the boxplots** also indicates the presence of demarcations or differences in the identification of likelihood of response variable. The notched box plots plotted with varying width was also considered in the likelihood of the significance of the varaibels.*
- ***Chi-square Test** was carried out for the features that had categories. This test statistic measures the significance with the response variable. Following was the hypothesis:*

  - *N0 : The feature is indepednent of the response variable*
  - *NA : The Feature is not independent of the response variable*

  ---

  *The p-value was considered to derive the correctness and applicability of the hypothesis. If all the other criteria were satisfactorilly met and the p-value for the chi-sq test < 0.05 , we concluded to have **strong evidence Against** the null hypothesis and we considered that features as potentially a good predictor. However, if the p-value > 0.05, we have **weak evidence against** the Null hypothesis and all these features were readily discarded.*

Considering all these Factors a bunch of 26 features were selected as a potential set of good predictors.

In [67]:

```
potential_predictors <- c(1,5,10,12,13,16,17,18,19,21,25,28,30,31,35,36,39,42,43
,44,45,46,47,48,49,50)
```

In [68]:

```
length(potential_predictors)
```

26

We also found features which exhibited higher revelance and correlation to the response in comparison to other featuers. This included :

1. *Contribution of Car Policies - PPERSAUT*
2. *Number of Car Policies - APERSAUT*
3. *Purchasing power class - MKOOPKLA*
4. *Number of fire policies - PBRAND*
5. *contribution to fire policies - ABRAND*
6. *Contribution of third party insurance - PWAPART*
7. *Number of third party insurances - AWPART*

However, we will try to arrive at these results by aforementioned methodology and approach. This will also serve as a confirmation for our hypothesis and observation.

Let us take a susbset of the features and reduce the number of features to 26

In [69]:

```
train_data_cop <- train_data_cop[,potential_predictors]
```

In [70]:

```
ncol(train_data_cop)
```

26

In [71]:

```
train_data_cop$X.CARAVAN = train_data$X.CARAVAN
```

In [72]:

```
ncol(train_data_cop)
```

27

We have subsetted the desired number of feautures . We will now proceed with some more filtering methods and perform application of models on these subset of features.

---

4. Identification of Highly Correlated Features

For Identification of highly correlated Features with the response, the norm is to use various coeffiecient methods to find how the variance in the predictor explains the variance in the response variable. The method being used differs depending on the type of data in the features. for eg

for two features with continuous values we use - ***Pearson's correlation coefficient*** *for two categorical variables we use -* ***Chi-sq Test***

Alternatively, when the dataset has a mixture of these varaibles, Logistic regression gives a good significance pertaining to the correlation of the features with the response variable. This seems to fit our case and therefore, we will go ahead with fitting a logistic regression model to our dataset and try to ascertain features that are highly correlated with the response variable.

In [73]:

```
model.glm <- glm(X.CARAVAN~.,data = train_data_cop)
```

In [74]:

```
summary(model.glm)
```

```
Call:
glm(formula = X.CARAVAN ~ ., data = train_data_cop)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.21266  -0.08898  -0.04956  -0.01037   1.02494

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.7698703  0.4076570   1.889 0.059005 .
X.MOSTYPE    0.0044679  0.0022124   2.020 0.043480 *
X.MOSHOOFD  -0.0203100  0.0098927  -2.053 0.040115 *
X.MRELGE     0.0104121  0.0035756   2.912 0.003605 **
X.MRELOV     0.0064778  0.0042501   1.524 0.127521
X.MFALLEEN  -0.0004617  0.0026703  -0.173 0.862735
X.MOPLHOOG   0.0009962  0.0063843   0.156 0.876002
X.MOPLMIDD  -0.0046363  0.0066782  -0.694 0.487553
X.MOPLLAAG  -0.0102834  0.0068796  -1.495 0.135034
X.MBERHOOG   0.0004338  0.0027173   0.160 0.873168
X.MBERBOER  -0.0079794  0.0035559  -2.244 0.024870 *
X.MSKA      -0.0009786  0.0031553  -0.310 0.756465
X.MSKC       0.0025871  0.0023351   1.108 0.267936
X.MHHUUR    -0.0447946  0.0365372  -1.226 0.220249
X.MHKOOP    -0.0423447  0.0365179  -1.160 0.246275
X.MZFONDS   -0.0446971  0.0436682  -1.024 0.306085
X.MZPART    -0.0473958  0.0436058  -1.087 0.277120
X.MINK.1    -0.0012500  0.0019617  -0.637 0.523992
X.MINKGEM    0.0046404  0.0031483   1.474 0.140561
X.MKOOPKLA   0.0034585  0.0021929   1.577 0.114825
X.PWAPART    0.0324232  0.0166149   1.951 0.051051 .
X.PPERSAUT   0.0090135  0.0026172   3.444 0.000577 ***
X.PBRAND     0.0123073  0.0034327   3.585 0.000339 ***
X.AWAPART   -0.0411131  0.0324314  -1.268 0.204958
X.APERSAUT   0.0084272  0.0125712   0.670 0.502654
X.ABROM     -0.0027267  0.0118988  -0.229 0.818754
X.ABRAND    -0.0219815  0.0115324  -1.906 0.056691 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 0.05365763)

    Null deviance: 327.20  on 5821  degrees of freedom
Residual deviance: 310.95  on 5795  degrees of freedom
AIC: -479.06

Number of Fisher Scoring iterations: 2
```

Throughout this Assessment, we will be using the approach of overlapping features. Just going by any one method and relying completely on it poses the threat of losing crucial information. Therefore, we will take the ***Overlapping*** *method wherein, the features that tend to overlap in both the methods. This will be essential in coming to a level playing field of including all possible variables that contribute towards the information and predictability of the response variable. Using the same method, we can close down on the number of subset of variables to be as follows :*

- *X.MOSTYPE*
- *X.MOSHOOFD*
- *X.MRELGE*
- *X.MBERBOER*
- *X.MKOOPKLA*
- *X.PWAPART*
- *X.PPERSAUT*
- *X.PBRAND*
- *X.ABRAND*
- *X.APERSAUT*
- *X.AWAPART*

However, we will try to use other wrapper methods and Feature Seletion methods to confirm these results and ensure that the principle of overlapping is maintained.

---

5. Wrapper Methods

In order to figure out the variable and confirm of the variables we chose conform to the prelimnary and Exploratory Data Analysis, We will apply Wrapper Methods to this dataset which will provide us with the information of significance of each of the variable and also we can confirm if these significant variables will help contribute towards a good reliable model. We will make use of Regsubset function and apply all possible methods : Backward, forward and exhaustive methods to see if the results provided are the same for each of the methods. We will also perform a accuracy check for these methods by applying various methods and checking the most accuracte model and subsequently use those features to predict Caravan Type

Let us first split the train_data_cop into train and test dataset with a 70:30 ratio.

In [75]:

```
set.seed(123)
trainIndex <- createDataPartition(y = train_data_cop$X.CARAVAN, p = 0.70, list =
FALSE)
train <- train_data_cop[trainIndex,]
test <- train_data_cop[-trainIndex,]
```

let us examine the size of the split dataset

In [76]:

```
nrow(train)
nrow(test)
```

4076

1746

The training data is split in 4076 for training dataset and 1746 for testing dataset

5.1 Subsetting Using RegSubsets

We will make use of the "Exhaustive" method to arrive at a subset of predictors first

In [77]:

```
regfit.full <- regsubsets(X.CARAVAN ~ .,train_data_cop, nvmax = ncol(train_data_
cop))
reg.summary <- summary(regfit.full)
reg.summary
```

```
Subset selection object
Call: regsubsets.formula(X.CARAVAN ~ ., train_data_cop, nvmax = ncol
(train_data_cop))
26 Variables  (and intercept)
           Forced in Forced out
X.MOSTYPE      FALSE      FALSE
X.MOSHOOFD     FALSE      FALSE
X.MRELGE       FALSE      FALSE
X.MRELOV       FALSE      FALSE
X.MFALLEEN     FALSE      FALSE
X.MOPLHOOG     FALSE      FALSE
X.MOPLMIDD     FALSE      FALSE
X.MOPLLAAG     FALSE      FALSE
X.MBERHOOG     FALSE      FALSE
X.MBERBOER     FALSE      FALSE
X.MSKA         FALSE      FALSE
X.MSKC         FALSE      FALSE
X.MHHUUR       FALSE      FALSE
X.MHKOOP       FALSE      FALSE
X.MZFONDS      FALSE      FALSE
X.MZPART       FALSE      FALSE
X.MINK.1       FALSE      FALSE
X.MINKGEM      FALSE      FALSE
X.MKOOPKLA     FALSE      FALSE
X.PWAPART      FALSE      FALSE
X.PPERSAUT     FALSE      FALSE
X.PBRAND       FALSE      FALSE
X.AWAPART      FALSE      FALSE
X.APERSAUT     FALSE      FALSE
X.ABROM        FALSE      FALSE
X.ABRAND       FALSE      FALSE
1 subsets of each size up to 26
Selection Algorithm: exhaustive
          X.MOSTYPE X.MOSHOOFD X.MRELGE X.MRELOV X.MFALLEEN X.MOPLHO
OG
1  ( 1 )  " "       " "        " "      " "      " "         " "
2  ( 1 )  " "       " "        " "      " "      " "         " "
3  ( 1 )  " "       " "        " "      " "      " "         " "
4  ( 1 )  " "       " "        "*"      " "      " "         " "
5  ( 1 )  " "       " "        "*"      " "      " "         " "
6  ( 1 )  " "       " "        "*"      " "      " "         " "
7  ( 1 )  " "       " "        "*"      " "      " "         " "
8  ( 1 )  " "       " "        "*"      " "      " "         " "
9  ( 1 )  " "       " "        "*"      " "      " "         " "
10  ( 1 ) " "       " "        "*"      " "      " "         " "
11  ( 1 ) " "       " "        "*"      " "      " "         " "
12  ( 1 ) " "       " "        "*"      "*"      " "         " "
13  ( 1 ) "*"       "*"        "*"      " "      " "         " "
14  ( 1 ) "*"       "*"        "*"      " "      " "         " "
15  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
16  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
17  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
18  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
19  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
20  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
21  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
22  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
23  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
24  ( 1 ) "*"       "*"        "*"      "*"      "*"         " "
25  ( 1 ) "*"       "*"        "*"      "*"      "*"         " "
26  ( 1 ) "*"       "*"        "*"      "*"      "*"         "*"
```

```
          X.MOPLMIDD X.MOPLLAAG X.MBERHOOG X.MBERBOER X.MSKA X.MSKC
X.MHHUUR
1  ( 1 )  " "        " "        " "        " "        " "    " "
" "
2  ( 1 )  " "        " "        " "        " "        " "    " "
" "
3  ( 1 )  " "        "*"        " "        " "        " "    " "
" "
4  ( 1 )  " "        "*"        " "        " "        " "    " "
" "
5  ( 1 )  " "        "*"        " "        "*"        " "    " "
" "
6  ( 1 )  " "        "*"        " "        "*"        " "    " "
" "
7  ( 1 )  " "        "*"        " "        "*"        " "    " "
"*"
8  ( 1 )  " "        "*"        " "        "*"        " "    " "
"*"
9  ( 1 )  "*"        "*"        " "        "*"        " "    " "
"*"
10  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
11  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
12  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
13  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
14  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
15  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
16  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
17  ( 1 ) "*"        "*"        " "        "*"        " "    " "
"*"
18  ( 1 ) "*"        "*"        " "        "*"        " "    "*"
"*"
19  ( 1 ) "*"        "*"        " "        "*"        " "    "*"
"*"
20  ( 1 ) "*"        "*"        " "        "*"        " "    "*"
"*"
21  ( 1 ) "*"        "*"        " "        "*"        " "    "*"
"*"
22  ( 1 ) "*"        "*"        " "        "*"        "*"    "*"
"*"
23  ( 1 ) "*"        "*"        " "        "*"        "*"    "*"
"*"
24  ( 1 ) "*"        "*"        " "        "*"        "*"    "*"
"*"
25  ( 1 ) "*"        "*"        "*"        "*"        "*"    "*"
"*"
26  ( 1 ) "*"        "*"        "*"        "*"        "*"    "*"
"*"
          X.MHKOOP X.MZFONDS X.MZPART X.MINK.1 X.MINKGEM X.MKOOPKLA
X.PWAPART
1  ( 1 )  " "       " "       " "      " "      " "       " "
" "
2  ( 1 )  " "       " "       " "      " "      " "       "*"
" "
3  ( 1 )  " "       " "       " "      " "      " "       " "
```

```
" "
4  ( 1 )  " "       " "       " "       " "       " "       " "
" "
5  ( 1 )  " "       " "       " "       " "       " "       " "
" "
6  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
7  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
8  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
9  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
10  ( 1 ) "*"       " "       " "       " "       " "       " "
"*"
11  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
12  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
13  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
14  ( 1 ) "*"       " "       "*"       " "       " "       "*"
"*"
15  ( 1 ) "*"       " "       "*"       " "       " "       "*"
"*"
16  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
17  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
18  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
19  ( 1 ) "*"       "*"       "*"       " "       "*"       "*"
"*"
20  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
21  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
22  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
23  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
24  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
25  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
26  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
          X.PPERSAUT X.PBRAND X.AWAPART X.APERSAUT X.ABROM X.ABRAND
1  ( 1 )  "*"          " "       " "         " "          " "      " "
2  ( 1 )  "*"          " "       " "         " "          " "      " "
3  ( 1 )  "*"          "*"       " "         " "          " "      " "
4  ( 1 )  "*"          "*"       " "         " "          " "      " "
5  ( 1 )  "*"          "*"       " "         " "          " "      " "
6  ( 1 )  "*"          "*"       " "         " "          " "      " "
7  ( 1 )  "*"          "*"       " "         " "          " "      " "
8  ( 1 )  "*"          "*"       " "         " "          " "      "*"
9  ( 1 )  "*"          "*"       " "         " "          " "      "*"
10  ( 1 ) "*"          "*"       " "         " "          " "      "*"
11  ( 1 ) "*"          "*"       " "         " "          " "      "*"
12  ( 1 ) "*"          "*"       " "         " "          " "      "*"
13  ( 1 ) "*"          "*"       " "         " "          " "      "*"
```

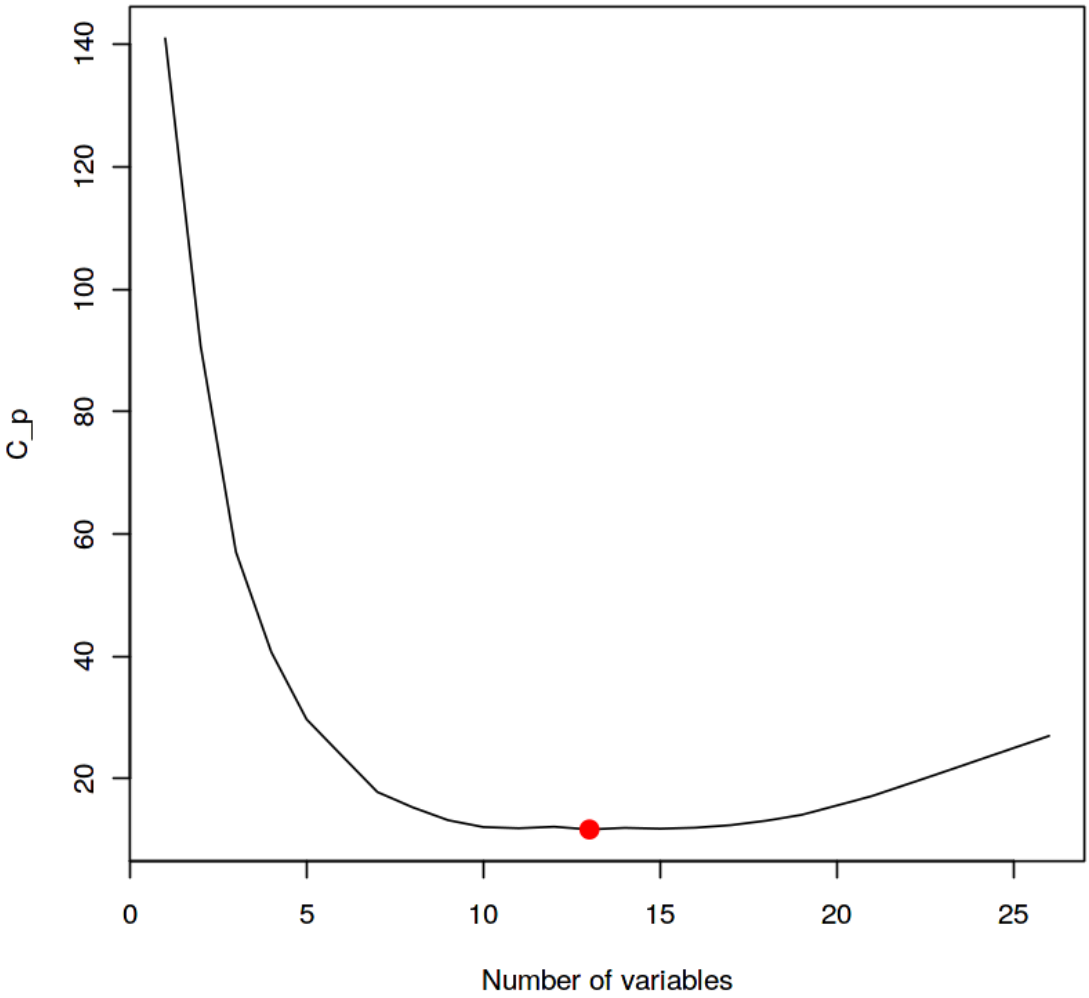
```
14  ( 1 ) "*"         "*"         " "         " "         " "         "*"
15  ( 1 ) "*"         "*"         " "         " "         " "         "*"
16  ( 1 ) "*"         "*"         " "         " "         " "         "*"
17  ( 1 ) "*"         "*"         "*"         " "         " "         "*"
18  ( 1 ) "*"         "*"         "*"         " "         " "         "*"
19  ( 1 ) "*"         "*"         "*"         " "         " "         "*"
20  ( 1 ) "*"         "*"         "*"         " "         " "         "*"
21  ( 1 ) "*"         "*"         "*"         "*"         " "         "*"
22  ( 1 ) "*"         "*"         "*"         "*"         " "         "*"
23  ( 1 ) "*"         "*"         "*"         "*"         "*"         "*"
24  ( 1 ) "*"         "*"         "*"         "*"         "*"         "*"
25  ( 1 ) "*"         "*"         "*"         "*"         "*"         "*"
26  ( 1 ) "*"         "*"         "*"         "*"         "*"         "*"
```

The performance of the subsets can be examined using several methods like : Mallow's cp, BIC, Adjusted R-squared value. These are model selection criteria for multiple regression model. We will perform each of these selection criteria on our subsetted featured model and choose the best out of it.

In [78]:

```
plot(reg.summary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
mincp = which.min(reg.summary$cp)
points(mincp, reg.summary$cp[mincp], col = "red", cex = 2, pch = 20)
mincp
```

13


C_p
20
40
60
80
100
120
140
0
5
10
15
20
25
Number of variables

Mallow's cp suggests that the model with 13 variables shows less variance and provides a good model. Let us find out what these varibales are.

In [79]:

```
x <- coef(regfit.full, mincp)
x
```

(Intercept)
0.586854100268068
X.MOSTYPE
0.0044875441197725
X.MOSHOOFD
-0.0202270879968155
X.MRELGE
0.00574300348693626
X.MOPLMIDD
-0.00451297670771434
X.MOPLLAAG
-0.00820781395596057
X.MBERBOER
-0.00959701875968222
X.MHHUUR
-0.0637513959057416
X.MHKOOP
-0.061350042184721
X.MKOOPKLA
0.00357078319873443
X.PWAPART
0.0121558003291304
X.PPERSAUT
0.0106900940719741
X.PBRAND
0.0127126915226695
X.ABRAND
-0.0238912289002154

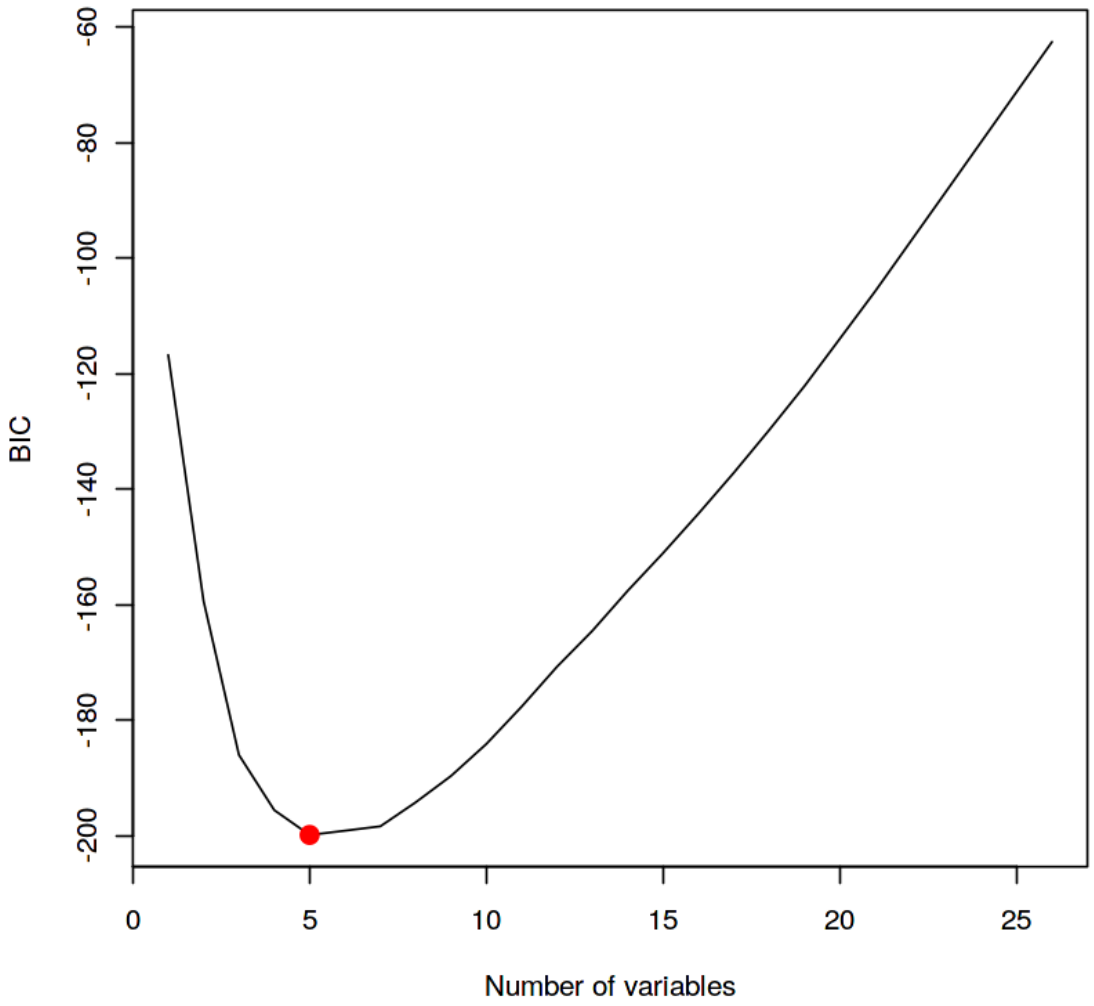
Let us check with other selelction criteria models like BIC and adjusted R-squared

In [80]:

```
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
minbic = which.min(reg.summary$bic)
points(minbic, reg.summary$bic[minbic], col = "red", cex = 2, pch = 20)
minbic
coef(regfit.full, minbic)
```

5

(Intercept)
0.00643640692910888
X.MRELGE
0.00700103017473284
X.MOPLLAAG
-0.00806945200157898
X.MBERBOER
-0.010502666733644
X.PPERSAUT
0.0113542863752852
X.PBRAND
0.0102325547660192



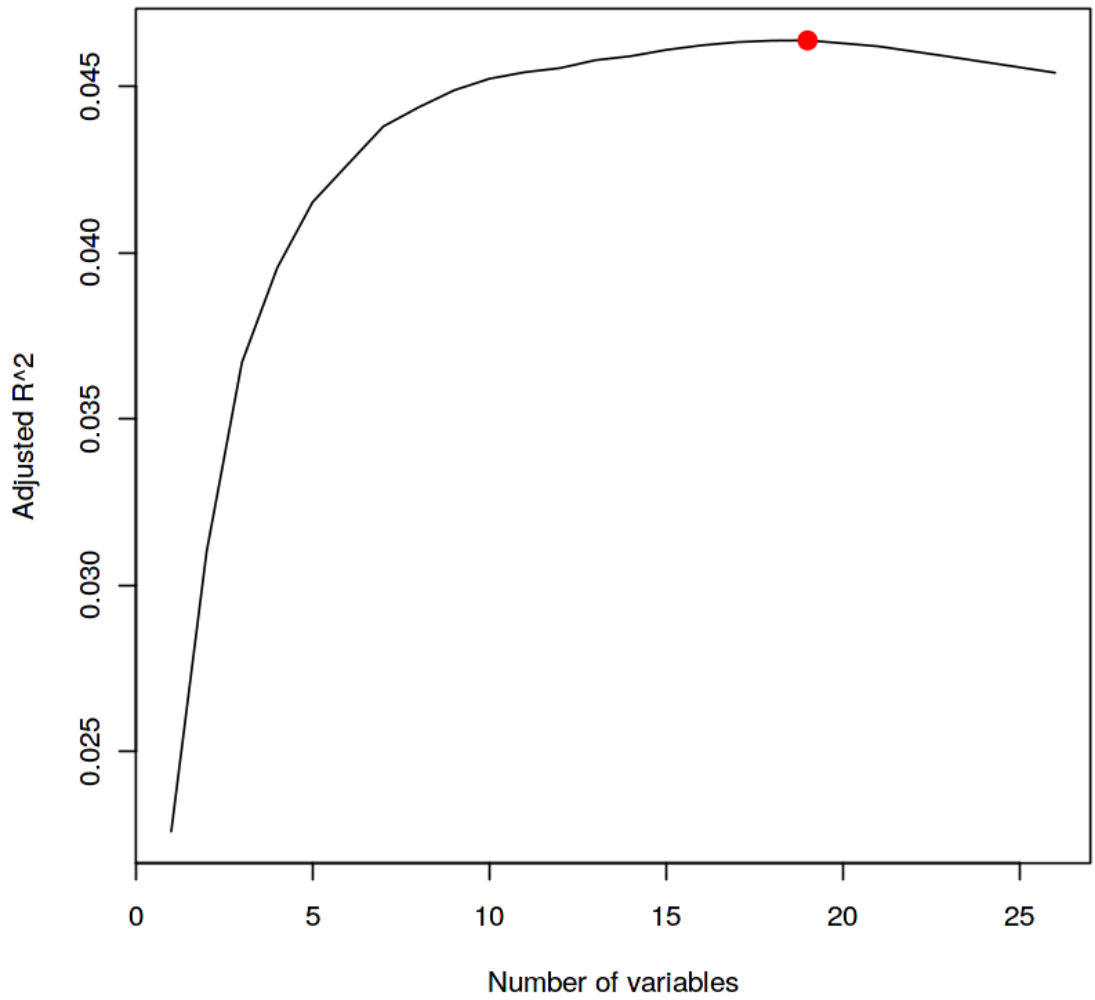
We can observe that we are getting conflicting results from both the methods. But we do have some features overlapping and that is what we are looking at currently. Let us apply adjusted R-squared method as well.

In [81]:

```
plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", typ
e = "l")
max_adjr2 = which.max(reg.summary$adjr2)
points(max_adjr2, reg.summary$adjr2[max_adjr2 ], col = "red", cex = 2, pch = 20)
max_adjr2
coef(regfit.full, max_adjr2)
```

19

(Intercept)
0.767596175243784
X.MOSTYPE
0.00447242024205285
X.MOSHOOFD
-0.0202816973395168
X.MRELGE
0.0104529307506049
X.MRELOV
0.00636617529681448
X.MOPLMIDD
-0.00505094191992979
X.MOPLLAAG
-0.0104805133946642
X.MBERBOER
-0.00850097382839702
X.MSKC
0.00251205264707364
X.MHHUUR
-0.0447514322592231
X.MHKOOP
-0.0423483569146955
X.MZFONDS
-0.0443919704221556
X.MZPART
-0.0471414627225265
X.MINKGEM
0.00407818592790099
X.MKOOPKLA
0.00348170124592328
X.PWAPART
0.0326193513783157
X.PPERSAUT
0.0106384939645316
X.PBRAND
0.0123665163417254
X.AWAPART
-0.0413907607739551
X.ABRAND
-0.0218412190222921



We can observe that we have achieved results that are far and wide. However, we can see that all the varibles we thought would be highly correlated to the response by looking at the Preliminary data anlaysis, Exploratory Data anslysis and other statistical tests, the same features appear in these results as well. Therefore we can go ahead and use those features to build our model. However, we can still perform one more set of validation of these features using the wrapper method which use K-Fold cross validation and selects a subset of features and does a forward subsetting of data. Before we do that, let us perform the similar operations as above to see if same results are achieved in "forward" and "Backward" feature selection.

5.1.2 Subsetting using Forward Selection

In [82]:

```
regfit.full <- regsubsets(X.CARAVAN ~ .,train_data_cop, nvmax = ncol(train_data_
cop),method="forward")
reg.summary <- summary(regfit.full)
reg.summary
```

```
Subset selection object
Call: regsubsets.formula(X.CARAVAN ~ ., train_data_cop, nvmax = ncol
(train_data_cop),
    method = "forward")
26 Variables  (and intercept)
           Forced in Forced out
X.MOSTYPE      FALSE      FALSE
X.MOSHOOFD     FALSE      FALSE
X.MRELGE       FALSE      FALSE
X.MRELOV       FALSE      FALSE
X.MFALLEEN     FALSE      FALSE
X.MOPLHOOG     FALSE      FALSE
X.MOPLMIDD     FALSE      FALSE
X.MOPLLAAG     FALSE      FALSE
X.MBERHOOG     FALSE      FALSE
X.MBERBOER     FALSE      FALSE
X.MSKA         FALSE      FALSE
X.MSKC         FALSE      FALSE
X.MHHUUR       FALSE      FALSE
X.MHKOOP       FALSE      FALSE
X.MZFONDS      FALSE      FALSE
X.MZPART       FALSE      FALSE
X.MINK.1       FALSE      FALSE
X.MINKGEM      FALSE      FALSE
X.MKOOPKLA     FALSE      FALSE
X.PWAPART      FALSE      FALSE
X.PPERSAUT     FALSE      FALSE
X.PBRAND       FALSE      FALSE
X.AWAPART      FALSE      FALSE
X.APERSAUT     FALSE      FALSE
X.ABROM        FALSE      FALSE
X.ABRAND       FALSE      FALSE
1 subsets of each size up to 26
Selection Algorithm: forward
          X.MOSTYPE X.MOSHOOFD X.MRELGE X.MRELOV X.MFALLEEN X.MOPLHO
OG
1  ( 1 )  " "       " "        " "      " "      " "         " "
2  ( 1 )  " "       " "        " "      " "      " "         " "
3  ( 1 )  " "       " "        " "      " "      " "         " "
4  ( 1 )  " "       " "        " "      " "      " "         " "
5  ( 1 )  " "       " "        "*"      " "      " "         " "
6  ( 1 )  " "       " "        "*"      " "      " "         " "
7  ( 1 )  " "       " "        "*"      " "      " "         " "
8  ( 1 )  " "       " "        "*"      " "      " "         " "
9  ( 1 )  " "       " "        "*"      " "      " "         " "
10  ( 1 ) " "       " "        "*"      " "      " "         " "
11  ( 1 ) " "       " "        "*"      " "      " "         " "
12  ( 1 ) " "       " "        "*"      "*"      " "         " "
13  ( 1 ) " "       " "        "*"      "*"      " "         " "
14  ( 1 ) " "       " "        "*"      "*"      " "         " "
15  ( 1 ) " "       " "        "*"      "*"      " "         " "
16  ( 1 ) " "       " "        "*"      "*"      " "         " "
17  ( 1 ) " "       " "        "*"      "*"      " "         " "
18  ( 1 ) " "       " "        "*"      "*"      " "         " "
19  ( 1 ) " "       " "        "*"      "*"      " "         " "
20  ( 1 ) " "       "*"        "*"      "*"      " "         " "
21  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
22  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
23  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
24  ( 1 ) "*"       "*"        "*"      "*"      "*"         " "
25  ( 1 ) "*"       "*"        "*"      "*"      "*"         " "
```

```
26  ( 1 ) "*"        "*"         "*"       "*"        "*"         "*"
          X.MOPLMIDD X.MOPLLAAG X.MBERHOOG X.MBERBOER X.MSKA X.MSKC
X.MHHUUR
1   ( 1 )  " "         " "         " "         " "         " "     " "
" "
2   ( 1 )  " "         " "         " "         " "         " "     " "
" "
3   ( 1 )  " "         " "         " "         " "         " "     " "
" "
4   ( 1 )  " "         "*"         " "         " "         " "     " "
" "
5   ( 1 )  " "         "*"         " "         " "         " "     " "
" "
6   ( 1 )  " "         "*"         " "         " "         " "     " "
" "
7   ( 1 )  " "         "*"         " "         "*"         " "     " "
" "
8   ( 1 )  " "         "*"         " "         "*"         " "     " "
"*"
9   ( 1 )  " "         "*"         " "         "*"         " "     " "
"*"
10  ( 1 )  " "         "*"         " "         "*"         " "     " "
"*"
11  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
12  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
13  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
14  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
15  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
16  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
17  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
18  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
19  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
20  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
21  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
22  ( 1 ) "*"         "*"         " "         "*"         "*"     "*"
"*"
23  ( 1 ) "*"         "*"         " "         "*"         "*"     "*"
"*"
24  ( 1 ) "*"         "*"         " "         "*"         "*"     "*"
"*"
25  ( 1 ) "*"         "*"         "*"         "*"         "*"     "*"
"*"
26  ( 1 ) "*"         "*"         "*"         "*"         "*"     "*"
"*"
          X.MHKOOP X.MZFONDS X.MZPART X.MINK.1 X.MINKGEM X.MKOOPKLA
X.PWAPART
1   ( 1 )  " "        " "         " "        " "        " "         " "
" "
2   ( 1 )  " "        " "         " "        " "        " "         "*"
" "
```

```
3  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
4  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
5  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
6  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
7  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
8  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
9  ( 1 )  " "       " "       " "       " "       " "       "*"
"*"
10  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
11  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
12  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
13  ( 1 ) "*"       " "       "*"       " "       " "       "*"
"*"
14  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
15  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
16  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
17  ( 1 ) "*"       "*"       "*"       " "       "*"       "*"
"*"
18  ( 1 ) "*"       "*"       "*"       " "       "*"       "*"
"*"
19  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
20  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
21  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
22  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
23  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
24  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
25  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
26  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
           X.PPERSAUT X.PBRAND X.AWAPART X.APERSAUT X.ABROM X.ABRAND
1  ( 1 )  "*"          " "       " "          " "          " "      " "
2  ( 1 )  "*"          " "       " "          " "          " "      " "
3  ( 1 )  "*"          " "       " "          " "          " "      " "
4  ( 1 )  "*"          " "       " "          " "          " "      " "
5  ( 1 )  "*"          " "       " "          " "          " "      " "
6  ( 1 )  "*"          "*"       " "          " "          " "      " "
7  ( 1 )  "*"          "*"       " "          " "          " "      " "
8  ( 1 )  "*"          "*"       " "          " "          " "      " "
9  ( 1 )  "*"          "*"       " "          " "          " "      "*"
10  ( 1 ) "*"          "*"       " "          " "          " "      "*"
11  ( 1 ) "*"          "*"       " "          " "          " "      "*"
12  ( 1 ) "*"          "*"       " "          " "          " "      "*"
```

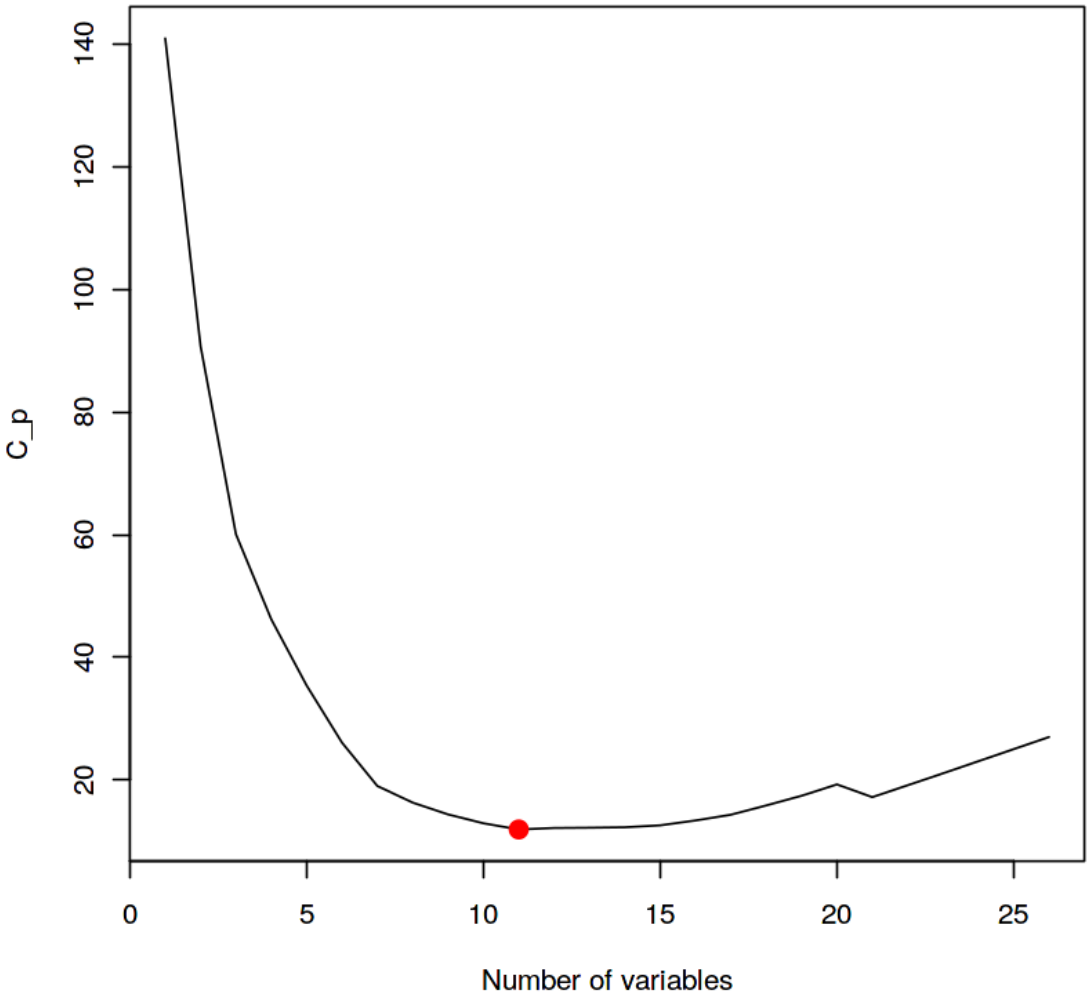
```
13  ( 1 ) "*"       "*"       " "       " "       " "       "*"
14  ( 1 ) "*"       "*"       " "       " "       " "       "*"
15  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
16  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
17  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
18  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
19  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
20  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
21  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
22  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
23  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
24  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
25  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
26  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
```

In [83]:

```
plot(reg.summary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
mincp = which.min(reg.summary$cp)
points(mincp, reg.summary$cp[mincp], col = "red", cex = 2, pch = 20)
mincp
coef(regfit.full,mincp)
```

11

(Intercept)
0.570170728576172
X.MRELGE
0.00490350951372999
X.MOPLMIDD
-0.00458838809782876
X.MOPLLAAG
-0.0085397273153856
X.MBERBOER
-0.0110688965667057
X.MHHUUR
-0.0615256795068896
X.MHKOOP
-0.0592108641796397
X.MKOOPKLA
0.0028183423667122
X.PWAPART
0.0126274497256253
X.PPERSAUT
0.0106430492702847
X.PBRAND
0.0122117970429345
X.ABRAND
-0.0226466424855639

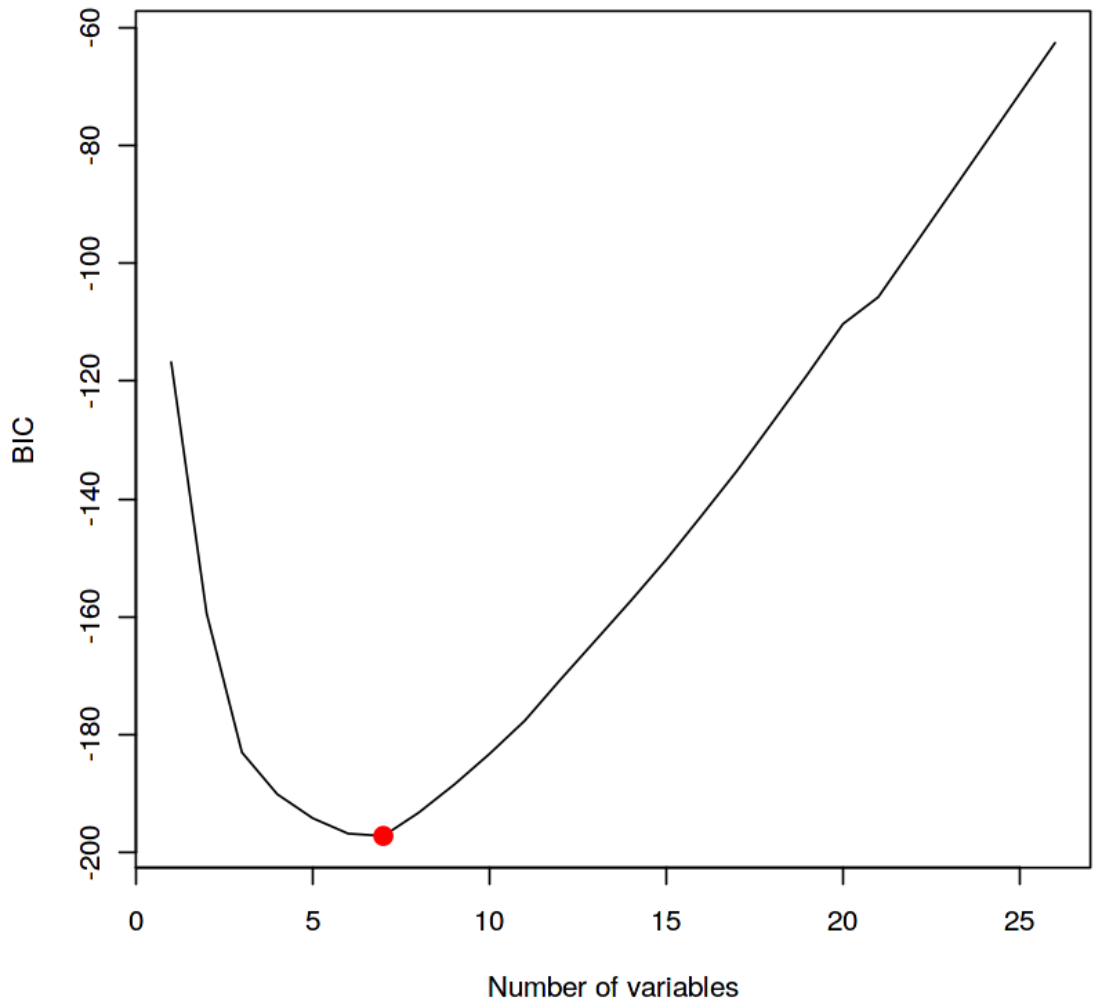

C_p
20
40
60
80
100
120
140
0
5
10
15
20
25
Number of variables

In [84]:

```
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
minbic = which.min(reg.summary$bic)
points(minbic, reg.summary$bic[minbic], col = "red", cex = 2, pch = 20)
minbic
coef(regfit.full, minbic)
```

7

(Intercept)
-0.0177664739722319
X.MRELGE
0.00596670060702019
X.MOPLLAAG
-0.00624253178475638
X.MBERBOER
-0.00886840419121445
X.MKOOPKLA
0.00468675346182456
X.PWAPART
0.0109453280999251
X.PPERSAUT
0.0109571293149773
X.PBRAND
0.0071003891059835

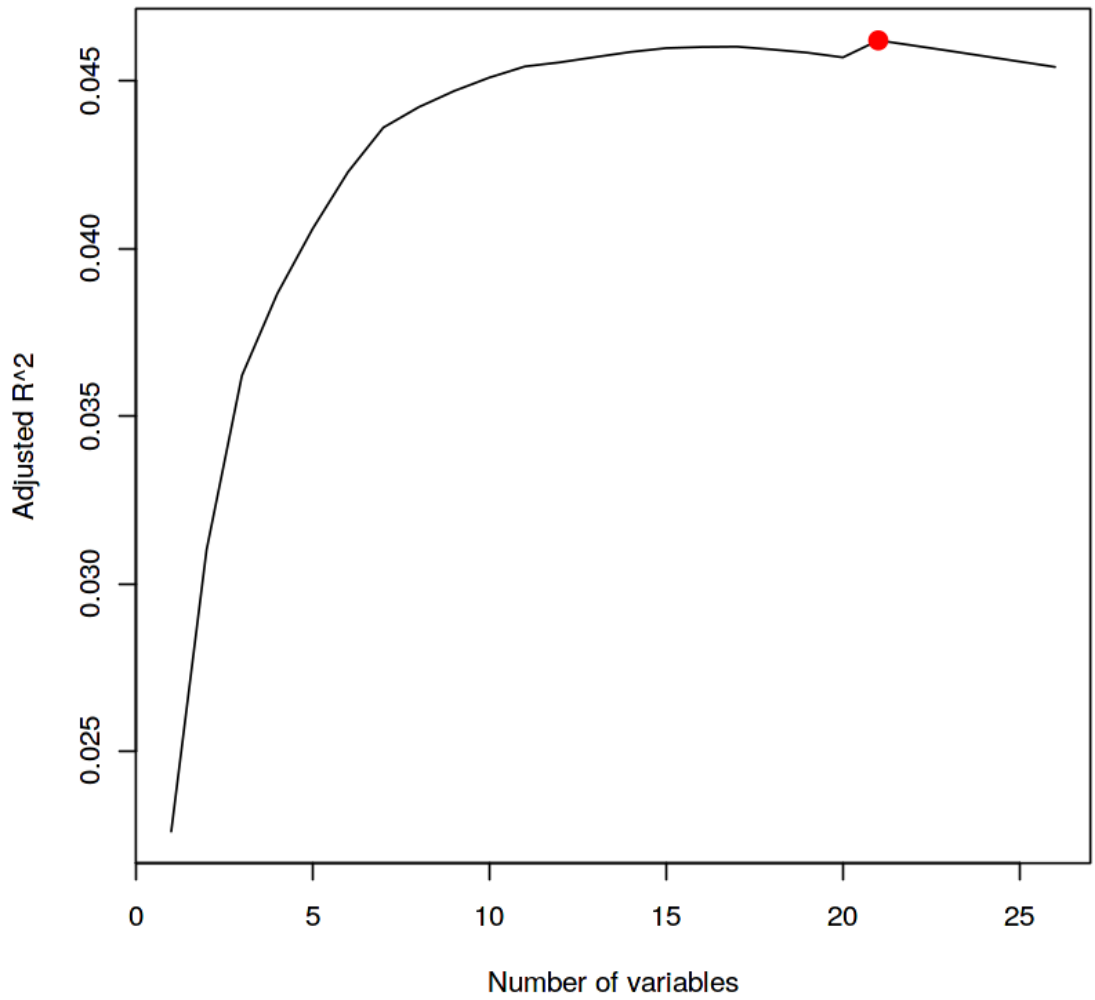


In [85]:

```
plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", typ
e = "l")
max_adjr2 = which.max(reg.summary$adjr2)
points(max_adjr2, reg.summary$adjr2[max_adjr2 ], col = "red", cex = 2, pch = 20)
max_adjr2
coef(regfit.full, max_adjr2)
```

21

(Intercept)
0.775174960690505
X.MOSTYPE
0.00446439280844128
X.MOSHOOFD
-0.0202903953668599
X.MRELGE
0.0103992276796135
X.MRELOV
0.00622198511155477
X.MOPLMIDD
-0.00521002946334661
X.MOPLLAAG
-0.010766915756449
X.MBERBOER
-0.00831198917078814
X.MSKC
0.00262330974146603
X.MHHUUR
-0.044660519282866
X.MHKOOP
-0.0421894400769417
X.MZFONDS
-0.0450294340308456
X.MZPART
-0.0477381513464863
X.MINK.1
-0.00130806916067772
X.MINKGEM
0.00477554900632062
X.MKOOPKLA
0.00350896475695832
X.PWAPART
0.0323881335817586
X.PPERSAUT
0.00906847198860278
X.PBRAND
0.0123019361368251
X.AWAPART
-0.040972258629962
X.APERSAUT
0.00837178929009614
X.ABRAND
-0.0217646228279537



We can see that the results do differ very minutely. However, we can still observe the overlapping of variables overall. Let us perform the same set of operations for backward subset selection.

5.1.3. Subsetting using Backward Selection

In [86]:

```
regfit.full <- regsubsets(X.CARAVAN ~ .,train_data_cop, nvmax = ncol(train_data_
cop),method="backward")
reg.summary <- summary(regfit.full)
reg.summary
```

```
Subset selection object
Call: regsubsets.formula(X.CARAVAN ~ ., train_data_cop, nvmax = ncol
(train_data_cop),
    method = "backward")
26 Variables  (and intercept)
           Forced in Forced out
X.MOSTYPE      FALSE      FALSE
X.MOSHOOFD     FALSE      FALSE
X.MRELGE       FALSE      FALSE
X.MRELOV       FALSE      FALSE
X.MFALLEEN     FALSE      FALSE
X.MOPLHOOG     FALSE      FALSE
X.MOPLMIDD     FALSE      FALSE
X.MOPLLAAG     FALSE      FALSE
X.MBERHOOG     FALSE      FALSE
X.MBERBOER     FALSE      FALSE
X.MSKA         FALSE      FALSE
X.MSKC         FALSE      FALSE
X.MHHUUR       FALSE      FALSE
X.MHKOOP       FALSE      FALSE
X.MZFONDS      FALSE      FALSE
X.MZPART       FALSE      FALSE
X.MINK.1       FALSE      FALSE
X.MINKGEM      FALSE      FALSE
X.MKOOPKLA     FALSE      FALSE
X.PWAPART      FALSE      FALSE
X.PPERSAUT     FALSE      FALSE
X.PBRAND       FALSE      FALSE
X.AWAPART      FALSE      FALSE
X.APERSAUT     FALSE      FALSE
X.ABROM        FALSE      FALSE
X.ABRAND       FALSE      FALSE
1 subsets of each size up to 26
Selection Algorithm: backward
          X.MOSTYPE X.MOSHOOFD X.MRELGE X.MRELOV X.MFALLEEN X.MOPLHO
OG
1  ( 1 )  " "       " "        " "      " "      " "         " "
2  ( 1 )  " "       " "        " "      " "      " "         " "
3  ( 1 )  " "       " "        " "      " "      " "         " "
4  ( 1 )  " "       " "        "*"      " "      " "         " "
5  ( 1 )  " "       " "        "*"      " "      " "         " "
6  ( 1 )  " "       " "        "*"      " "      " "         " "
7  ( 1 )  " "       " "        "*"      " "      " "         " "
8  ( 1 )  " "       " "        "*"      " "      " "         " "
9  ( 1 )  " "       " "        "*"      " "      " "         " "
10  ( 1 ) " "       " "        "*"      " "      " "         " "
11  ( 1 ) " "       "*"        "*"      " "      " "         " "
12  ( 1 ) "*"       "*"        "*"      " "      " "         " "
13  ( 1 ) "*"       "*"        "*"      " "      " "         " "
14  ( 1 ) "*"       "*"        "*"      " "      " "         " "
15  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
16  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
17  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
18  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
19  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
20  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
21  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
22  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
23  ( 1 ) "*"       "*"        "*"      "*"      " "         " "
24  ( 1 ) "*"       "*"        "*"      "*"      "*"         " "
25  ( 1 ) "*"       "*"        "*"      "*"      "*"         " "
```

```
26  ( 1 ) "*"        "*"         "*"       "*"        "*"         "*"
          X.MOPLMIDD X.MOPLLAAG X.MBERHOOG X.MBERBOER X.MSKA X.MSKC
X.MHHUUR
1   ( 1 )  " "         " "         " "         " "         " "     " "
" "
2   ( 1 )  " "         "*"         " "         " "         " "     " "
" "
3   ( 1 )  " "         "*"         " "         " "         " "     " "
" "
4   ( 1 )  " "         "*"         " "         " "         " "     " "
" "
5   ( 1 )  " "         "*"         " "         "*"         " "     " "
" "
6   ( 1 )  " "         "*"         " "         "*"         " "     " "
" "
7   ( 1 )  " "         "*"         " "         "*"         " "     " "
"*"
8   ( 1 )  " "         "*"         " "         "*"         " "     " "
"*"
9   ( 1 )  "*"         "*"         " "         "*"         " "     " "
"*"
10  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
11  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
12  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
13  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
14  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
15  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
16  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
17  ( 1 ) "*"         "*"         " "         "*"         " "     " "
"*"
18  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
19  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
20  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
21  ( 1 ) "*"         "*"         " "         "*"         " "     "*"
"*"
22  ( 1 ) "*"         "*"         " "         "*"         "*"     "*"
"*"
23  ( 1 ) "*"         "*"         " "         "*"         "*"     "*"
"*"
24  ( 1 ) "*"         "*"         " "         "*"         "*"     "*"
"*"
25  ( 1 ) "*"         "*"         "*"         "*"         "*"     "*"
"*"
26  ( 1 ) "*"         "*"         "*"         "*"         "*"     "*"
"*"
          X.MHKOOP X.MZFONDS X.MZPART X.MINK.1 X.MINKGEM X.MKOOPKLA
X.PWAPART
1   ( 1 )  " "        " "         " "        " "        " "         " "
" "
2   ( 1 )  " "        " "         " "        " "        " "         " "
" "
```

```
3  ( 1 )  " "       " "       " "       " "       " "       " "
" "
4  ( 1 )  " "       " "       " "       " "       " "       " "
" "
5  ( 1 )  " "       " "       " "       " "       " "       " "
" "
6  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
7  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
8  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
9  ( 1 )  " "       " "       " "       " "       " "       " "
"*"
10  ( 1 ) "*"       " "       " "       " "       " "       " "
"*"
11  ( 1 ) "*"       " "       " "       " "       " "       " "
"*"
12  ( 1 ) "*"       " "       " "       " "       " "       " "
"*"
13  ( 1 ) "*"       " "       " "       " "       " "       "*"
"*"
14  ( 1 ) "*"       " "       "*"       " "       " "       "*"
"*"
15  ( 1 ) "*"       " "       "*"       " "       " "       "*"
"*"
16  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
17  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
18  ( 1 ) "*"       " "       "*"       " "       "*"       "*"
"*"
19  ( 1 ) "*"       "*"       "*"       " "       "*"       "*"
"*"
20  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
21  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
22  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
23  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
24  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
25  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
26  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
"*"
           X.PPERSAUT X.PBRAND X.AWAPART X.APERSAUT X.ABROM X.ABRAND
1  ( 1 )  "*"          " "       " "          " "          " "      " "
2  ( 1 )  "*"          " "       " "          " "          " "      " "
3  ( 1 )  "*"          "*"       " "          " "          " "      " "
4  ( 1 )  "*"          "*"       " "          " "          " "      " "
5  ( 1 )  "*"          "*"       " "          " "          " "      " "
6  ( 1 )  "*"          "*"       " "          " "          " "      " "
7  ( 1 )  "*"          "*"       " "          " "          " "      " "
8  ( 1 )  "*"          "*"       " "          " "          " "      "*"
9  ( 1 )  "*"          "*"       " "          " "          " "      "*"
10  ( 1 ) "*"          "*"       " "          " "          " "      "*"
11  ( 1 ) "*"          "*"       " "          " "          " "      "*"
12  ( 1 ) "*"          "*"       " "          " "          " "      "*"
```

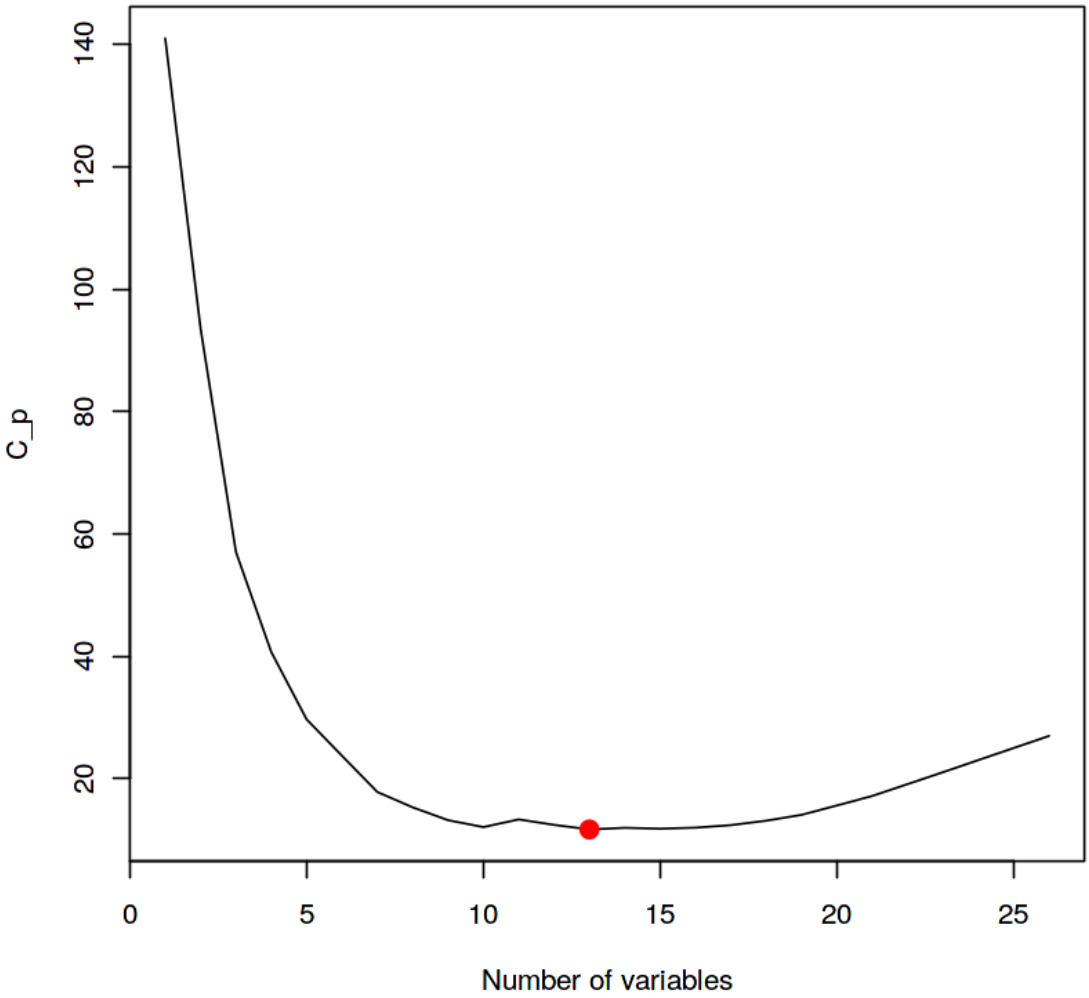
```
13  ( 1 ) "*"       "*"       " "       " "       " "       "*"
14  ( 1 ) "*"       "*"       " "       " "       " "       "*"
15  ( 1 ) "*"       "*"       " "       " "       " "       "*"
16  ( 1 ) "*"       "*"       " "       " "       " "       "*"
17  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
18  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
19  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
20  ( 1 ) "*"       "*"       "*"       " "       " "       "*"
21  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
22  ( 1 ) "*"       "*"       "*"       "*"       " "       "*"
23  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
24  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
25  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
26  ( 1 ) "*"       "*"       "*"       "*"       "*"       "*"
```

In [87]:

```
plot(reg.summary$cp, xlab = "Number of variables", ylab = "C_p", type = "l")
mincp = which.min(reg.summary$cp)
points(mincp, reg.summary$cp[mincp], col = "red", cex = 2, pch = 20)
mincp
coef(regfit.full,mincp)
```

13

(Intercept)
0.586854100257692
X.MOSTYPE
0.00448754412016234
X.MOSHOOFD
-0.0202270879980323
X.MRELGE
0.00574300348663686
X.MOPLMIDD
-0.00451297670763807
X.MOPLLAAG
-0.00820781395570462
X.MBERBOER
-0.00959701875951135
X.MHHUUR
-0.0637513959054873
X.MHKOOP
-0.0613500421847557
X.MKOOPKLA
0.00357078320056716
X.PWAPART
0.012155800328204
X.PPERSAUT
0.0106900940719366
X.PBRAND
0.0127126915224925
X.ABRAND
-0.0238912288988904

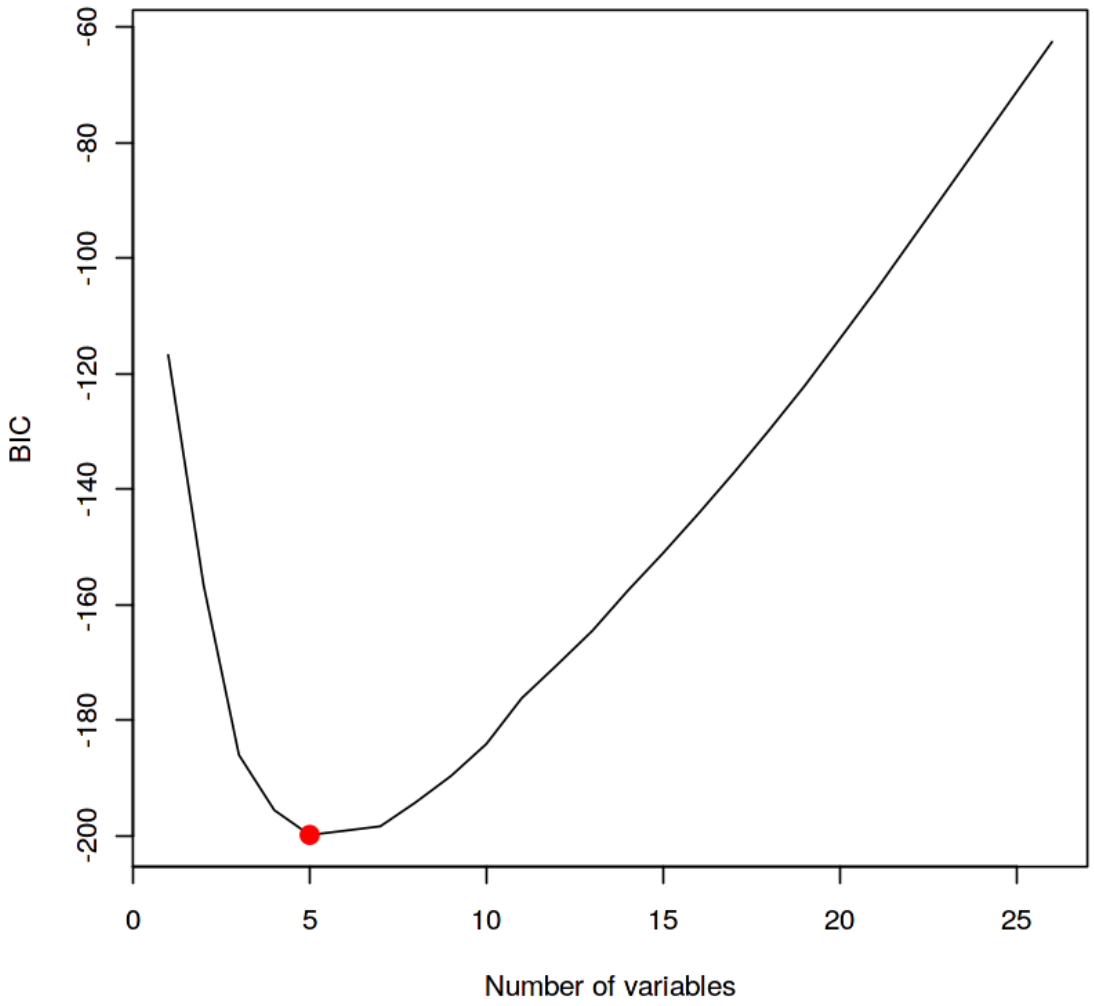

C_p
20
40
60
80
100
120
140
0
5
10
15
20
25
Number of variables

In [88]:

```
plot(reg.summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
minbic = which.min(reg.summary$bic)
points(minbic, reg.summary$bic[minbic], col = "red", cex = 2, pch = 20)
minbic
coef(regfit.full, minbic)
```

5

(Intercept)
0.00643640692910652
X.MRELGE
0.00700103017473313
X.MOPLLAAG
-0.00806945200157897
X.MBERBOER
-0.0105026667336436
X.PPERSAUT
0.0113542863752854
X.PBRAND
0.0102325547660192

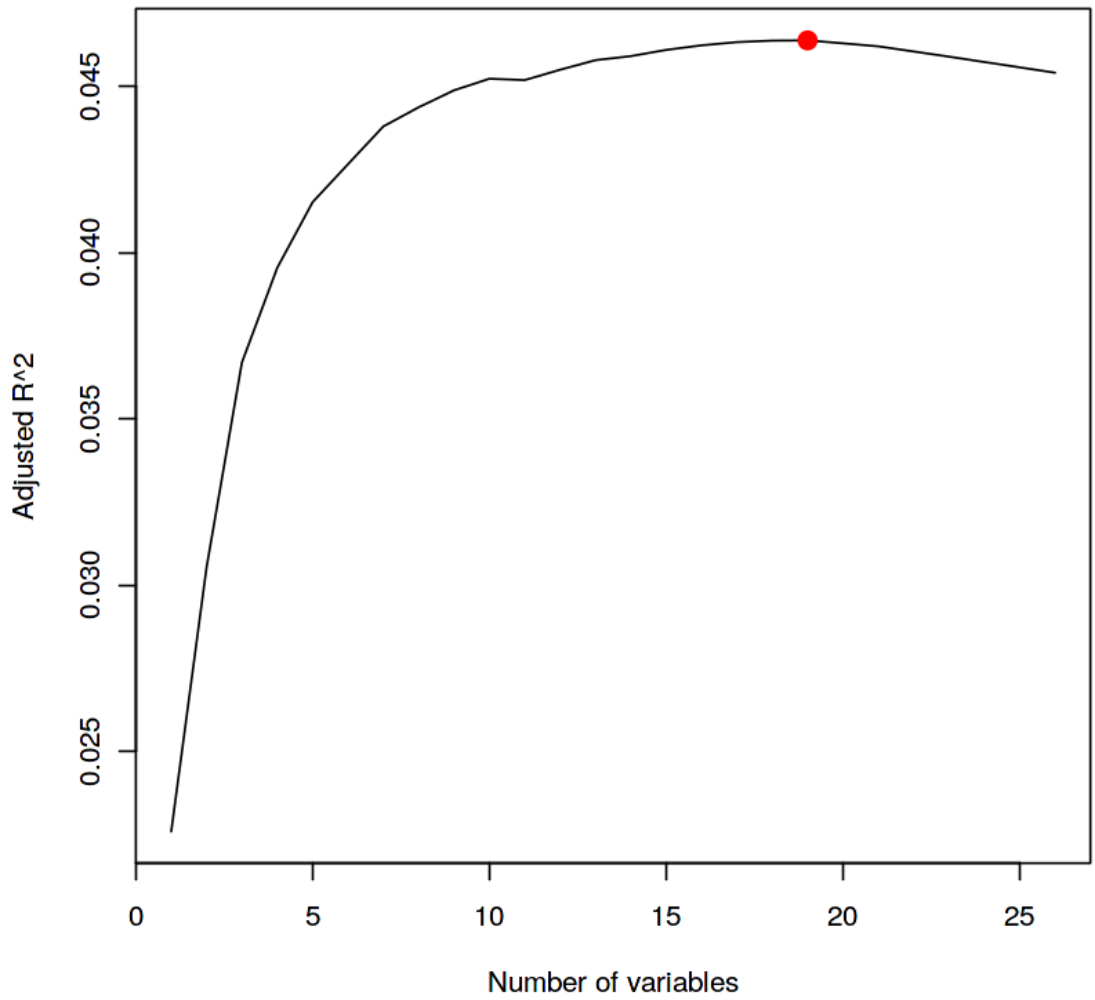


In [89]:

```
plot(reg.summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R^2", typ
e = "l")
max_adjr2 = which.max(reg.summary$adjr2)
points(max_adjr2, reg.summary$adjr2[max_adjr2 ], col = "red", cex = 2, pch = 20)
max_adjr2
coef(regfit.full, max_adjr2)
```

19

(Intercept)
0.767596175244368
X.MOSTYPE
0.00447242024204929
X.MOSHOOFD
-0.0202816973395332
X.MRELGE
0.0104529307506066
X.MRELOV
0.00636617529681813
X.MOPLMIDD
-0.00505094191992671
X.MOPLLAAG
-0.0104805133947161
X.MBERBOER
-0.00850097382846039
X.MSKC
0.00251205264707193
X.MHHUUR
-0.0447514322592593
X.MHKOOP
-0.0423483569146998
X.MZFONDS
-0.0443919704221421
X.MZPART
-0.0471414627225154
X.MINKGEM
0.00407818592788889
X.MKOOPKLA
0.00348170124590736
X.PWAPART
0.0326193513589463
X.PPERSAUT
0.0106384939645321
X.PBRAND
0.0123665163419926
X.AWAPART
-0.041390760737066
X.ABRAND
-0.0218412190229944



After performing all methods of subset selection we can see that there are a selected few features that appear in all the susbet selection criteria models. Therefore, we have a fair playing groud of number of variables that can be used in further modelling processes. before we move on to building our first model, let us perform a forward selection using k-fold cross validation just to cement our Preliminary, Exploratory and Susbet selection processes.

5.2 Foward Selection using K-fold Cross Validation

In [90]:

```
fold <- function(d, k = 10) {

    # The number of rows in *d*.
    r <- nrow(d)

    # The number of rows per fold.
    n <- r %/% k

    # Minimally "over-sample" from the values 1 to *k*,
    # by "shuffling" *n* + 1 repetitions of the values 1 to *k*,
    # and "take" *r* values (to account for an uneven division of *r*).
    id <- sample(rep(1:k, n + 1))[1:r]

    return(id)
}
```

In [93]:

```
# Cross-validate AUC for the predictors specified in *f*,
# for a logistic regression model, for the data *d*.
# Parameters:
#   d: data (data frame)
#   f: formula specifying the predictors
#   k: number of folds (default 10)
#   s: random seed (default 1)
# Returns:
#   list:
#     - AUC
#     - AUC standard error
#     - ROC curve (roc object)
xv <- function(d, f, k = 10, s = 1) {

    # Set the random seed.
    set.seed(s)

    # Create *k* cross-validation folds for the data *d*.
    id <- fold(d, k)

    # Initialise the set of all prediction probabilities,
    # for all *k* cross-validation folds.
    p_ <- NULL

    # Initialise the set of all "true" class values,
    # for all *k* cross-validation folds.
    t_ <- NULL

    # For each of *k* folds:
    for (i in 1:k) {

        # Define the training set.
        tr <- d[i != id, ]

        # Define the test set.
        te <- d[i == id, ]

        # Fit a logistic regression model for the specified predictors.
        m <- glm(f, tr, family = binomial)

        # Apply the model to the test set.
        p <- predict(m, te, type = "response")

        # Accumulate the prediction probabilities.
        p_ <- c(p_, p)

        # Accumulate the "true" class values.
        t_ <- c(t_, as.vector(te$X.CARAVAN))
    }

    # Compute the ROC curve for the set of all prediction probabilities.
    r <- roc(t_, p_)

    # Compute AUC.
    a <- pROC::auc(r)

    # Compute AUC standard error.
    se <- sqrt(var(r))
```

```
    return(list("AUC" = a, "SE" = se, "ROC" = r))
}
```

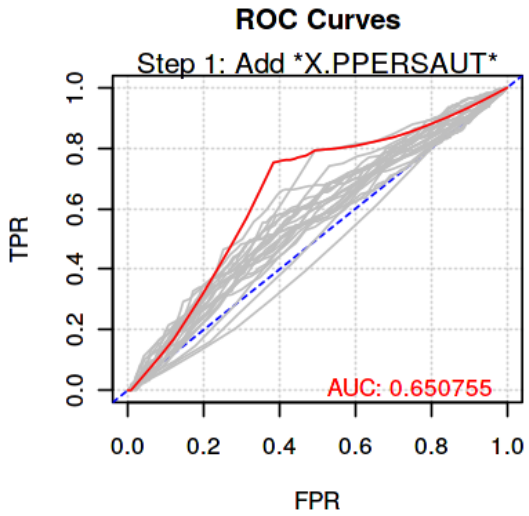
In [94]:

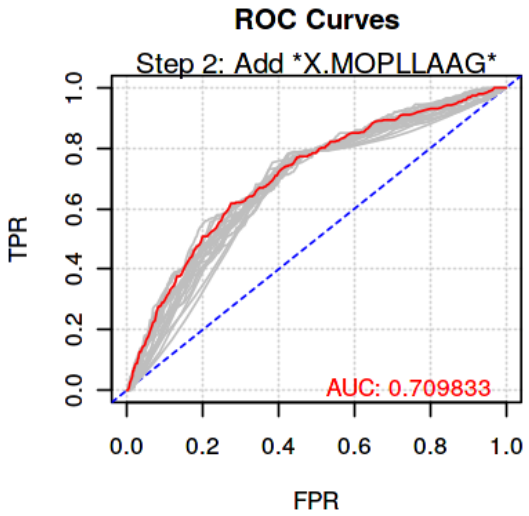
```
# Perform forward stepwise selection using cross-validated AUC,
# for a logistic regression model, for the data *d*.
# Parameters:
#   d: data (data frame)
# Returns:
#   best (data frame):
#     - step number
#     - predictor name
#     - AUC
#     - AUC standard error
step_xv <- function(d) {

    # The set of predictors.
    x <- colnames(d[1:(ncol(d) - 1)])

    # The response.
    y <- colnames(d[ncol(d)])

    # The number of predictors.
    n <- length(x)

    # Initialise the return value (data frame).
    best <- data.frame("Step" = 1:n, "Var" = "", "AUC" = 0, "SE" = 0,
                       stringsAsFactors = FALSE)

    # For each of *n* predictors:
    for (i in 1:n) {

        # Start a new plot.
        plot.new()
        plot.window(xlim = c(1, 0), ylim = c(0, 1))
        grid()
        box()
        abline(1, -1, col = "blue", lty = 2)
        l <- seq(0,1,0.2)
        axis(1, at = rev(l), labels = format(l, 2))
        axis(2)
        title(main = "ROC Curves", xlab = "FPR", ylab = "TPR")

        # The number of predictors not yet in the model.
        m <- length(x)

        # For each of *m* predictors not yet in the model:
        for (j in 1:m) {

            # Define the formula for the current model plus the predictor.
            f <- paste(y, paste(c(best$Var[1:i], x[j]), collapse = " + "),
                       sep = " ~ ")
            f <- formula(f)
            # Cross-validate AUC for the candidate model.
            xv_j <- xv(d, f)

            # Plot the ROC curve.
            plot(xv_j$ROC, col = "grey", lwd = 1, add = TRUE)

            # If the AUC is larger than the current largest AUC,
            # update the best candidate model.
            if (xv_j$AUC > best$AUC[i]) {
                best$AUC[i] <- xv_j$AUC
```

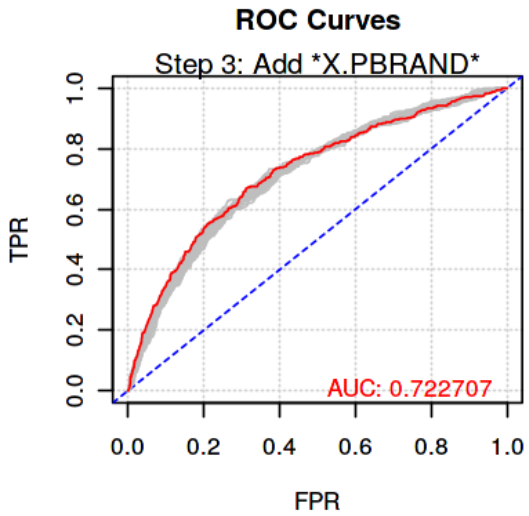
```
                best$SE[i] <- xv_j$SE
                best_r <- xv_j$ROC
                best_j <- j
            }
        }

        # Add the best predictor to the current model.
        best$Var[i] <- x[best_j]

        # Remove the best predictor,
        # from the set of predictors not yet in the model.
        x <- x[-best_j]

        # Plot the ROC curve for the new model.
        plot(best_r, col = "red", lwd = 1, add = TRUE)
        text(0, 0, paste("AUC: ", round(best$AUC[i], 6), sep = ""),
             pos = 2, col = "red")
        mtext(paste("Step ", i, ": Add *", best$Var[i], "*", sep = ""))
    }

    return(best)
}
```

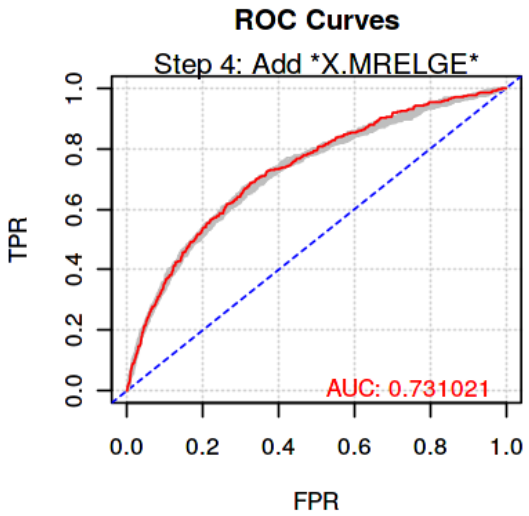
In [96]:

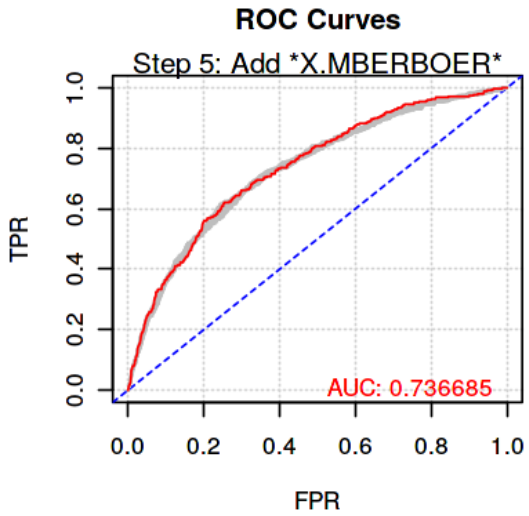
```
par(mfrow = c(2, 2))

# Perform stepwise selection by cross-validated AUC,
# for a logistic regression model, for the data *d*.
st <- step_xv(train_data_cop)
```

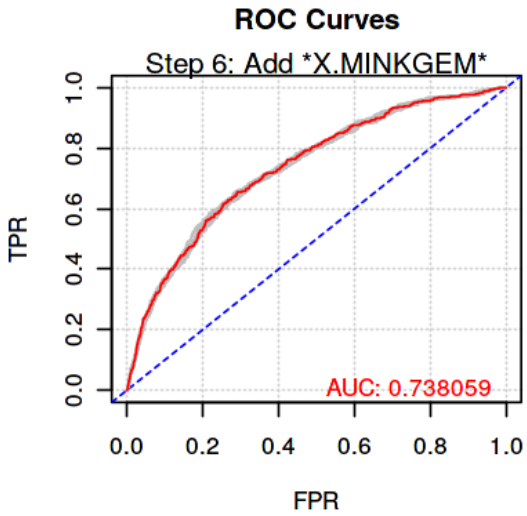

ROC Curves
Step 1: Add *X.PPERSAUT*
TPR
FPR
AUC: 0.650755
0.0
0.2
0.4
0.6
0.8
1.0

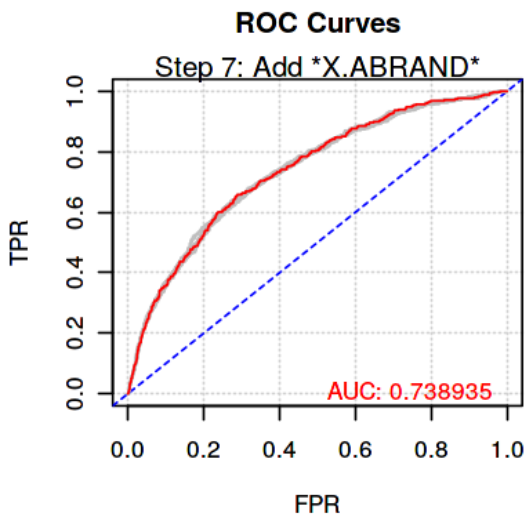

ROC Curves
Step 2: Add *X.MOPLLAAG*
TPR
FPR
AUC: 0.709833
0.0
0.2
0.4
0.6
0.8
1.0

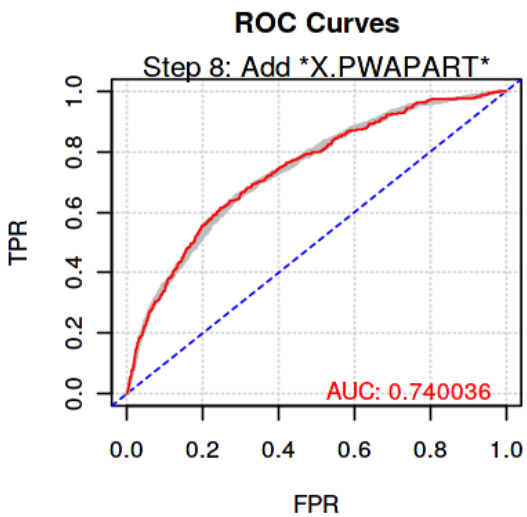

ROC Curves
Step 3: Add *X.PBRAND*
TPR
FPR
AUC: 0.722707
0.0
0.2
0.4
0.6
0.8
1.0

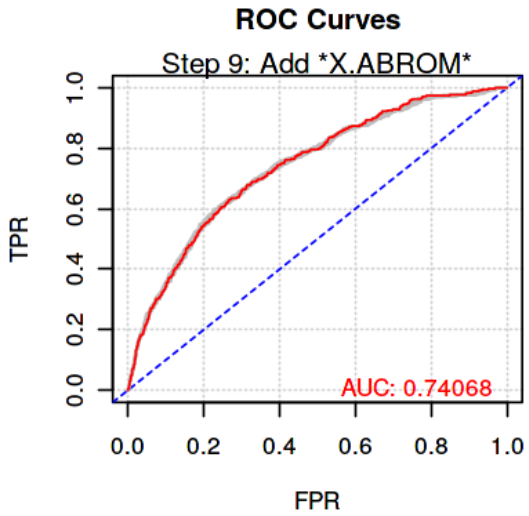

ROC Curves
Step 4: Add *X.MRELGE*
TPR
FPR
AUC: 0.731021
0.0
0.2
0.4
0.6
0.8
1.0

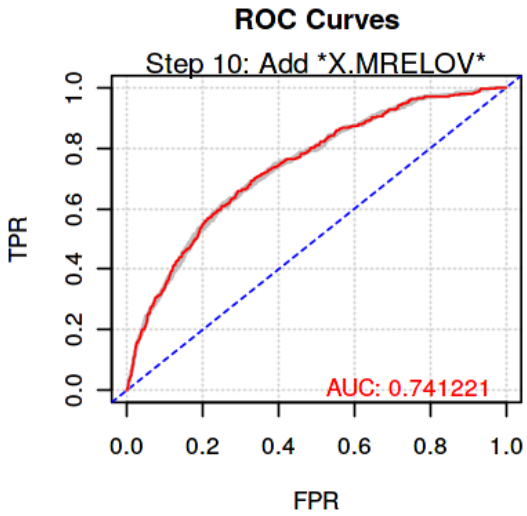

ROC Curves
Step 5: Add *X.MBERBOER*
TPR
FPR
AUC: 0.736685
0.0
0.2
0.4
0.6
0.8
1.0

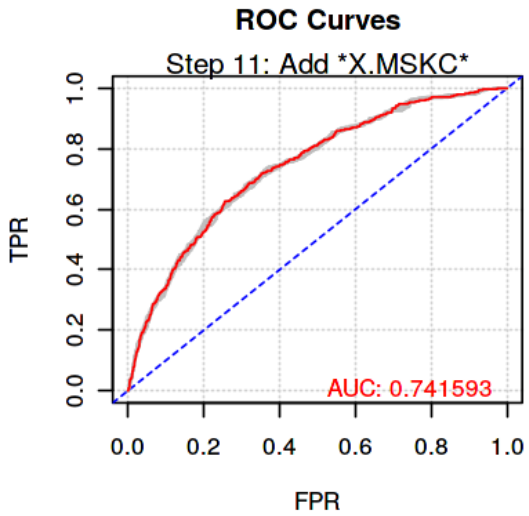

ROC Curves
Step 6: Add *X.MINKGEM*
TPR
FPR
AUC: 0.738059
0.0
0.2
0.4
0.6
0.8
1.0

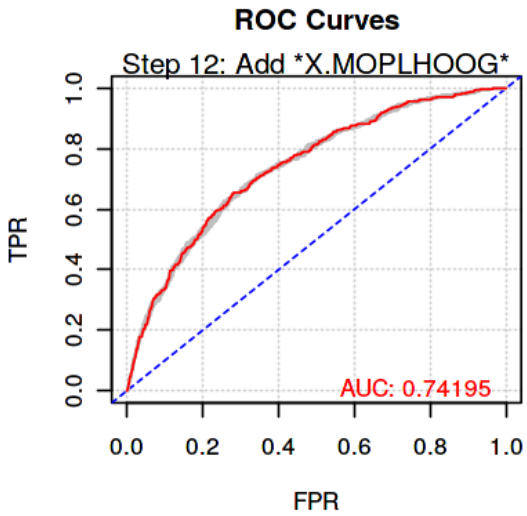

ROC Curves
Step 7: Add *X.ABRAND*
TPR
FPR
AUC: 0.738935
0.0
0.2
0.4
0.6
0.8
1.0

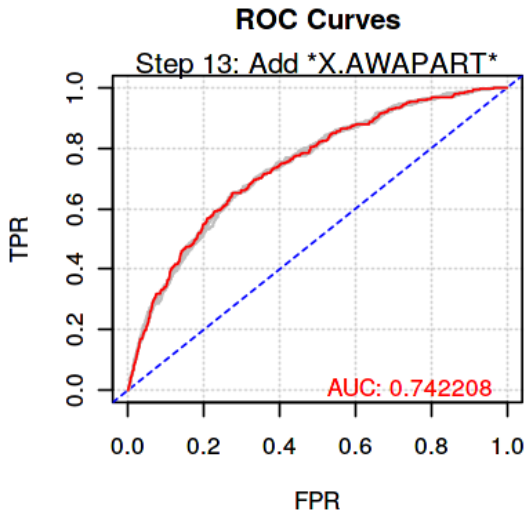

ROC Curves
Step 8: Add *X.PWAPART*
TPR
FPR
AUC: 0.740036
0.0
0.2
0.4
0.6
0.8
1.0

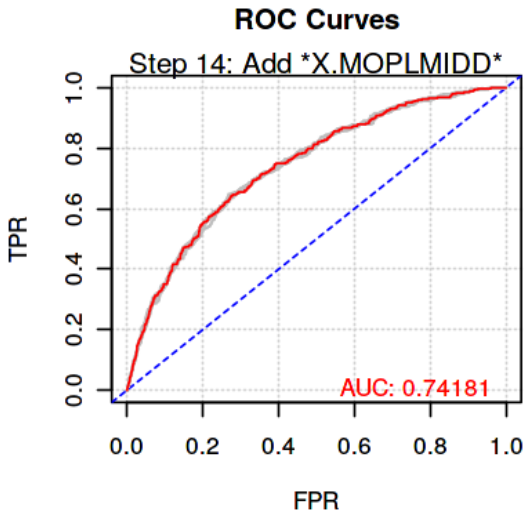

ROC Curves
Step 9: Add *X.ABROM*
TPR
FPR
AUC: 0.74068
0.0
0.2
0.4
0.6
0.8
1.0

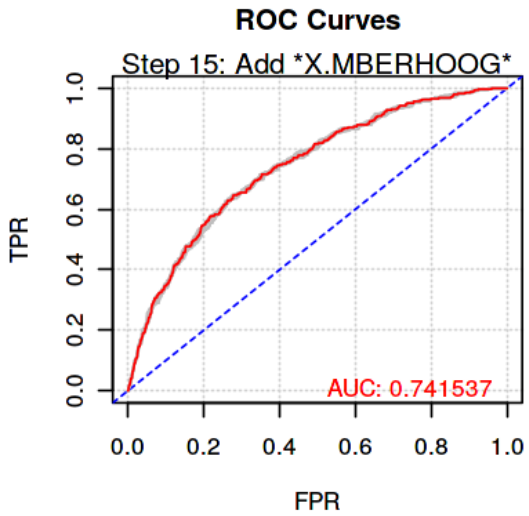

ROC Curves
Step 10: Add *X.MRELOV*
TPR
FPR
AUC: 0.741221
0.0
0.2
0.4
0.6
0.8
1.0

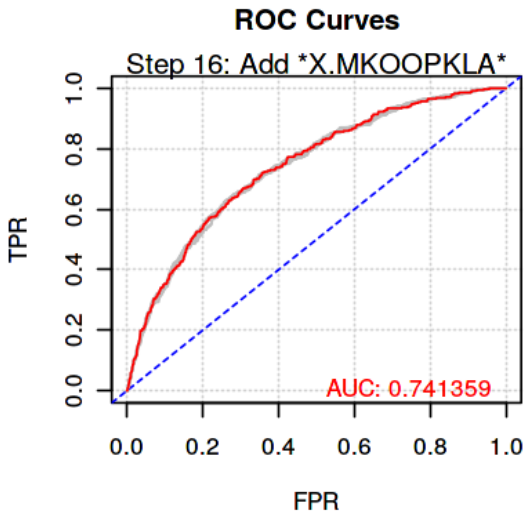

ROC Curves
Step 11: Add *X.MSKC*
TPR
FPR
AUC: 0.741593
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curves
Step 12: Add *X.MOPLHOOG*
TPR
FPR
AUC: 0.74195
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curves
Step 13: Add *X.AWAPART*
TPR
FPR
AUC: 0.742208
0.0
0.2
0.4
0.6
0.8
1.0

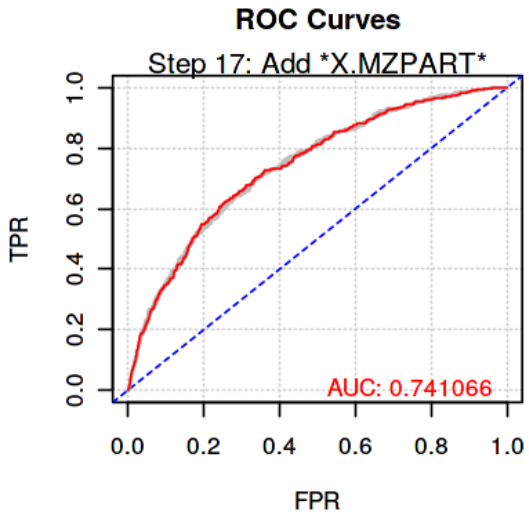

ROC Curves
Step 14: Add *X.MOPLMIDD*
TPR
FPR
AUC: 0.74181
0.0
0.2
0.4
0.6
0.8
1.0

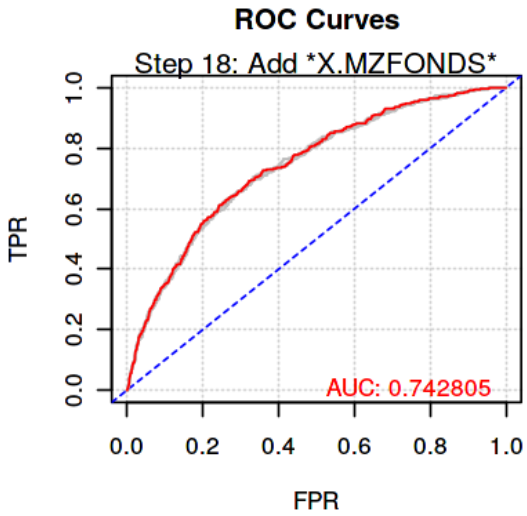

ROC Curves
Step 15: Add *X.MBERHOOG*
TPR
FPR
AUC: 0.741537
0.0
0.2
0.4
0.6
0.8
1.0

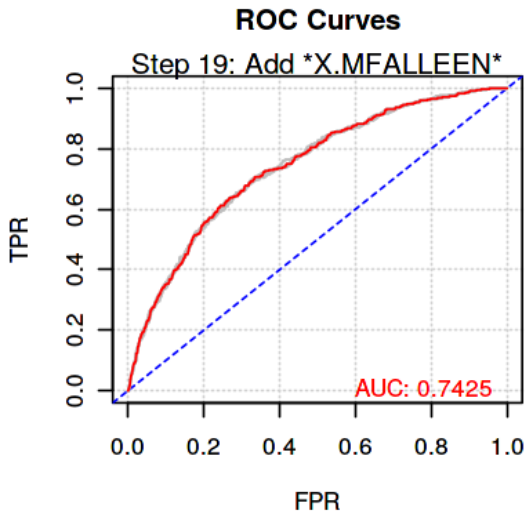

ROC Curves
Step 16: Add *X.MKOOPKLA*
TPR
FPR
AUC: 0.741359
0.0
0.2
0.4
0.6
0.8
1.0

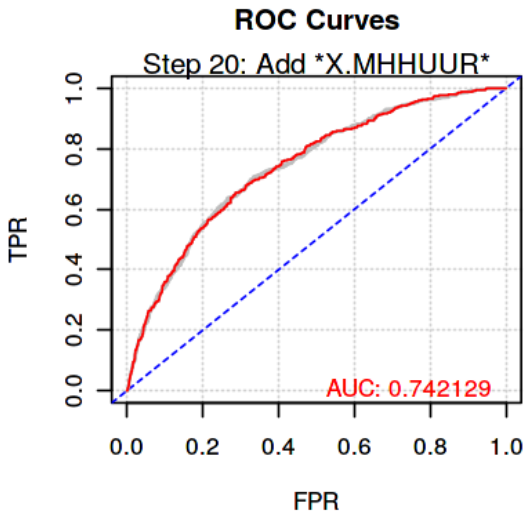
```
Warning message:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
```

```
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”Warning m
essage:
“glm.fit: fitted probabilities numerically 0 or 1 occurred”
```

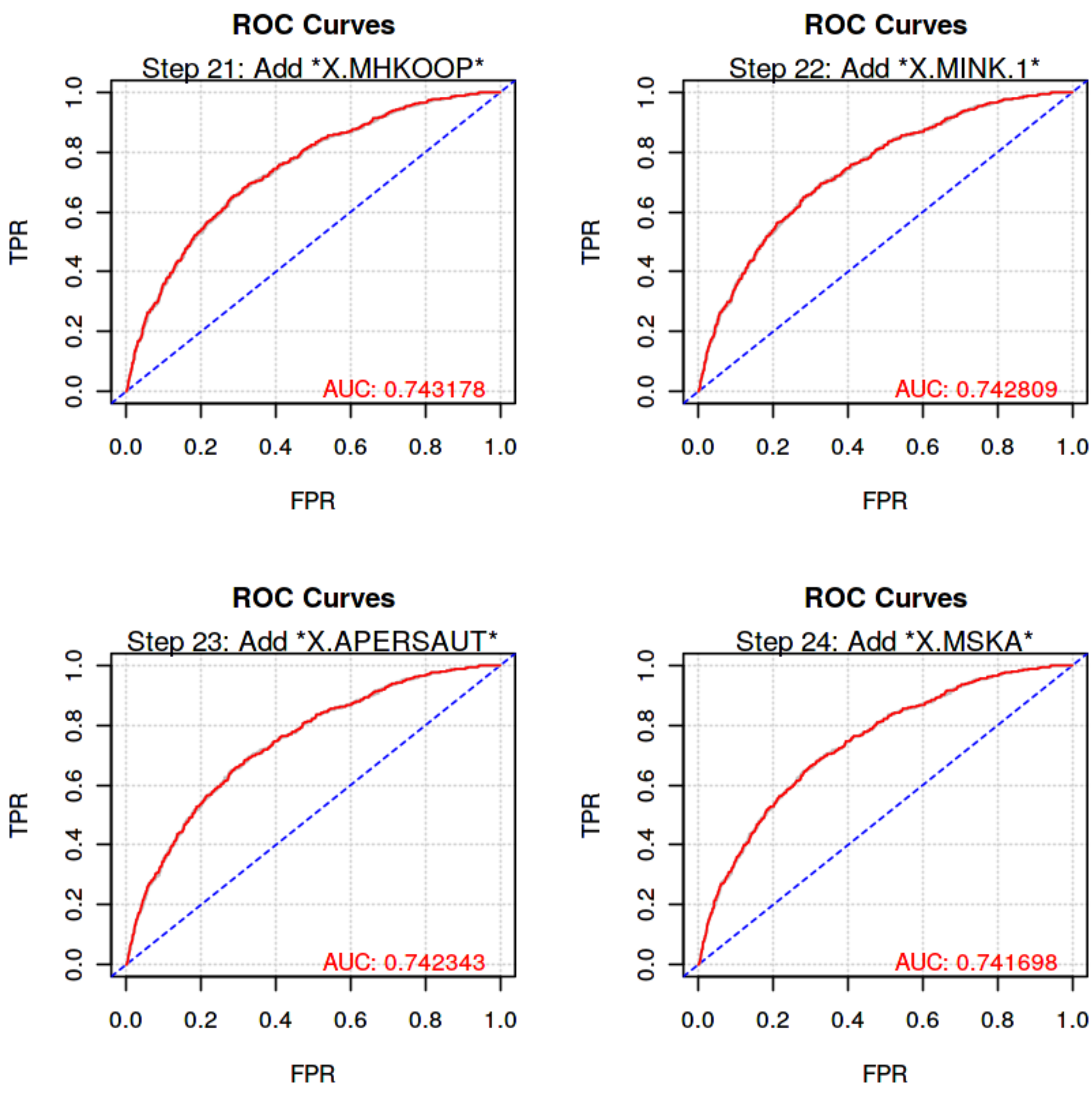

ROC Curves
Step 17: Add *X.MZPART*
TPR
FPR
AUC: 0.741066
0.0
0.2
0.4
0.6
0.8
1.0

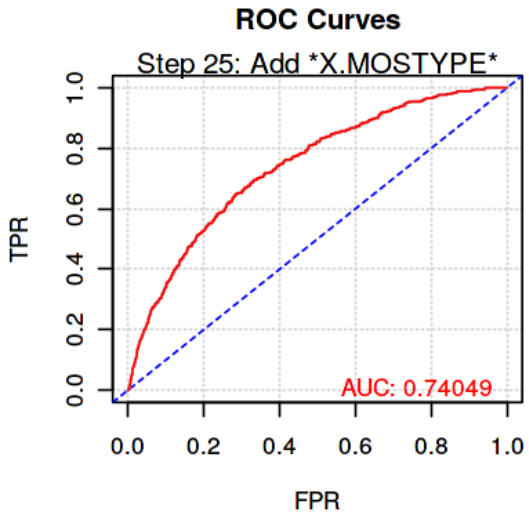

ROC Curves
Step 18: Add *X.MZFONDS*
TPR
FPR
AUC: 0.742805
0.0
0.2
0.4
0.6
0.8
1.0

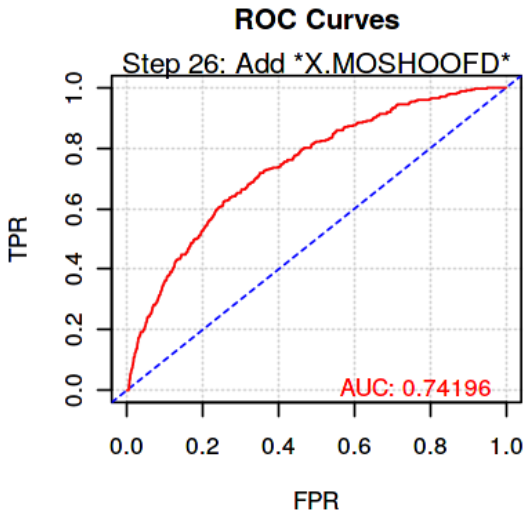

ROC Curves
Step 19: Add *X.MFALLEEN*
TPR
FPR
AUC: 0.7425
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curves
Step 20: Add *X.MHHUUR*
TPR
FPR
AUC: 0.742129
0.0
0.2
0.4
0.6
0.8
1.0

```
Warning message:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"Warning m
essage:
"glm.fit: fitted probabilities numerically 0 or 1 occurred"
```







The way this function works is that it picks a random subset of data from the dataset and adds a predictor, builds a logistic regression model on it, calculates the AUC for the highest AUC Value of the corresponding predictor. The first predictor added the is the single best predictor that provides a good level of Specificity for the response variable. in the subsequent iterations, it adds the variable which increases the AUC of the model and keeps adding the features to the model until all the predictors of the dataset have been exhausted.

Now that we have performed K-fold cross validation for the selection of subsets of features from the dataset, let us start intepreting the ROC and AUC Curves.

***ROC Curve(Reciever Operating Characteristic)** : shows the performance of a classification model at all possible thresholds. Decreasing the threshold increases the rate of False positives and True positives . the key is to find the optimum threshold which balances the performance of the model.*

***AUC Curve(Area under the curve)** : The step that exhibits highest AUC score represents the best subset of predictors. closer the AUC value to 1 the better is the model. The grey plot shows the performace of all possible predictors considered for that step and the red plot shows ROC curve with the highest AUC value/largest increase in AUC.*

Let us Analyze the AUC and SE for each of the predictors.

In [97]:

```
st
```

Step	Var	AUC	SE
1	X.PPERSAUT	0.6507552	0.01473410
2	X.MOPLLAAG	0.7098334	0.01439628
3	X.PBRAND	0.7227072	0.01462913
4	X.MRELGE	0.7310213	0.01412381
5	X.MBERBOER	0.7366852	0.01380775
6	X.MINKGEM	0.7380590	0.01377602
7	X.ABRAND	0.7389354	0.01363590
8	X.PWAPART	0.7400357	0.01367902
9	X.ABROM	0.7406796	0.01342330
10	X.MRELOV	0.7412210	0.01343473
11	X.MSKC	0.7415932	0.01340363
12	X.MOPLHOOG	0.7419497	0.01336686
13	X.AWAPART	0.7422079	0.01341979
14	X.MOPLMIDD	0.7418098	0.01342624
15	X.MBERHOOG	0.7415365	0.01341530
16	X.MKOOPKLA	0.7413588	0.01343137
17	X.MZPART	0.7410664	0.01346301
18	X.MZFONDS	0.7428051	0.01336900
19	X.MFALLEEN	0.7425001	0.01336930
20	X.MHHUUR	0.7421295	0.01333915
21	X.MHKOOP	0.7431778	0.01327455
22	X.MINK.1	0.7428093	0.01327439
23	X.APERSAUT	0.7423431	0.01328384
24	X.MSKA	0.7416980	0.01327637
25	X.MOSTYPE	0.7404900	0.01327987
26	X.MOSHOOFD	0.7419604	0.01332839

In [98]:

```
# Return the model (step) with the highest AUC value.
st[st$AUC == max(st$AUC), ]
```

	Step	Var	AUC	SE
21	21	X.MHKOOP	0.7431778	0.01327455

In [99]:

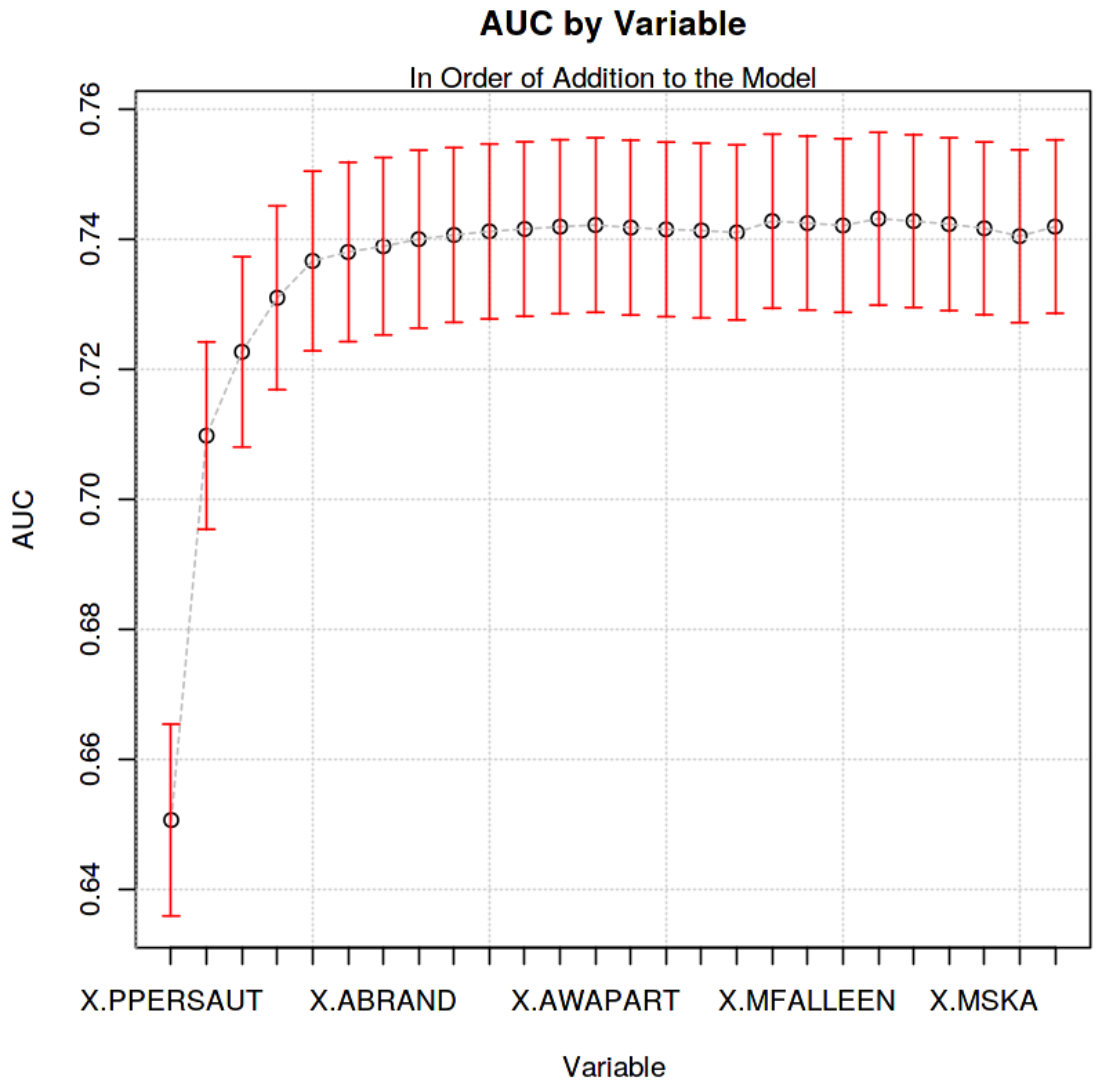
```
# Return the models (steps) within one standard error of the AUC value,
# of the model with the the highest AUC value.
st[st$AUC >= max(st$AUC) - st$SE[st$AUC == max(st$AUC)], ]
```

	Step	Var	AUC	SE
4	4	X.MRELGE	0.7310213	0.01412381
5	5	X.MBERBOER	0.7366852	0.01380775
6	6	X.MINKGEM	0.7380590	0.01377602
7	7	X.ABRAND	0.7389354	0.01363590
8	8	X.PWAPART	0.7400357	0.01367902
9	9	X.ABROM	0.7406796	0.01342330
10	10	X.MRELOV	0.7412210	0.01343473
11	11	X.MSKC	0.7415932	0.01340363
12	12	X.MOPLHOOG	0.7419497	0.01336686
13	13	X.AWAPART	0.7422079	0.01341979
14	14	X.MOPLMIDD	0.7418098	0.01342624
15	15	X.MBERHOOG	0.7415365	0.01341530
16	16	X.MKOOPKLA	0.7413588	0.01343137
17	17	X.MZPART	0.7410664	0.01346301
18	18	X.MZFONDS	0.7428051	0.01336900
19	19	X.MFALLEEN	0.7425001	0.01336930
20	20	X.MHHUUR	0.7421295	0.01333915
21	21	X.MHKOOP	0.7431778	0.01327455
22	22	X.MINK.1	0.7428093	0.01327439
23	23	X.APERSAUT	0.7423431	0.01328384
24	24	X.MSKA	0.7416980	0.01327637
25	25	X.MOSTYPE	0.7404900	0.01327987
26	26	X.MOSHOOFD	0.7419604	0.01332839

We can see that there is a slight increase in the AUC with each value. However, adding all these variables will add many variables to the model. therefore , making it more susceptible to overfitting. Also the AUC gets stagnant after reaching certain value. That value should be considered a cut-off. The above variables are indicative of the variables which have have 1 std deviation from the highest AUC Value of a predictor.

In [100]:

```
# Calculate the minimum and maximum y (AUC) values.
min_y <- min(st$AUC) - max(st$SE)
max_y <- max(st$AUC) + max(st$SE)

# Plot the AUC values for each step.
plot(st$AUC, xlab = "Variable", ylab = "AUC", main = "AUC by Variable",
     xaxt = "n", ylim = c(min_y, max_y))
grid()
mtext("In Order of Addition to the Model")
axis(1, at = 1:26, labels = st$Var)
lines(st$AUC, col = "grey", lty = 2)

# Show bars (modified arrows) for the AUC standard error for each step.
# Source: <stackoverflow.com/questions/13032777/scatter-plot-with-error-bars>.
arrows(st$Step, st$AUC - st$SE, st$Step, st$AUC + st$SE, code = 3, angle = 90,
       length = 0.05, col = "red")
```



We can observe that after step 13 the AUC value does not change by much. Therefore, we can consider the features appearing in the first 14 features and find a subset of features which overlap with all our previous analysis.

In [149]:

```
st$Var[1:13]
```

'X.PPERSAUT' 'X.MOPLLAAG' 'X.PBRAND' 'X.MRELGE' 'X.MBERBOER' 'X.MINKGEM' 'X.ABRAND' 'X.PWAPART' 'X.ABROM' 'X.MRELOV' 'X.MSKC' 'X.MOPLHOOG' 'X.AWAPART'

After Performing All the the preliminary Anlaysis, Exploratory Data anlaysis, Statisitcal Tests, Feature selection and finding subsets, K fold cross validation for dimensionality reduction, we can conclude that the features which we conclude to be highly correlated to the response variable as the best overallping featurs in almost all the tests. therefore, we will now perform Cross Validation and build our model. The final Set of Features selected for modeling will be :

- *X.MKOOPKLA*
- *X.PWAPART*
- *X.PPERSAUT*
- *X.PBRAND*
- *X.ABRAND*
- *X.APERSAUT*
- *X.AWAPART*

---

6. Cross Validation and Model Performance Evaluation

*In this stage of the problem solution, we will cross validation on the several models. The models that are chosen will be proabailitic Models as we are trying to find the **likelihood** of a customer buying a Caravn policy. Considering the response variable to be a categorical feature, we can perform the following Models:*

- ***Logisitic Regression Model** : The First model that could suit the dataset is Logistic regression model. the fact that the response variable has 2 categories and Logisitic regression works ideally with response variables with 2 classes, we can fit the model on the training set using a base model: using the features suggested by the forward step approach and the overlapping features and compare the model performance.*
- ***Naive Bayes classifier** : Another model that helps classify a response variable into Categories. we can make use of the posterior probabilities to predict the categories of the response variable. Although a simplisitic model, can potentially provide a reliable solution to a classification problem. We will perform cross validation using a set of features as the base model and then perform classification on the overallping features and compare the model performances.*
- ***Linear discrimanant Analysis** : A model that also works on the posterior probabilities for a feature with categorical type. It is often used as a means to dimensionality reduction and is popularly regarded as a filtering method alongside PCA(Principal Component Analysis). It tries to separate a multi-dimensional data into two separate classes by maximising the mean differences between the features by projecting the fearures on a 2-dimesnional space. We can work with the posterior probabilities to come up with probabilities for the test data.*

6.1 Logistic regression

let us perform logistic regression on the dataset , which is already split into test and train

6.1.1 Logisitic Regression with Suggested Features

Let us make use of the caret package which provides cross validation for logistic regression

In [111]:

```
train_control <- trainControl(method = "cv", number = 10,savePredictions = TRUE)
```

In [155]:

```
model <- train(as.factor(X.CARAVAN) ~ X.PPERSAUT+X.MOPLLAAG+X.PBRAND+X.MRELGE+X.
MBERBOER+X.MINKGEM+X.ABRAND+X.PWAPART+X.ABROM+X.MRELOV+X.MSKC+X.MOPLHOOG+X.AWAPA
RT,
               data = train,
               trControl = train_control,
               method = "glm",
               family=binomial())

# print cv scores
summary(model)

pred = as.data.frame(predict(model, newdata=test,type="prob"))
pred$new_prob <- ifelse(pred$'1' > 0.2,1,0)
print("------------------------------------------------------------")
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Call:
NULL

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.0603  -0.3974  -0.2725  -0.1853   3.0424

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.43661    0.79586  -6.831 8.43e-12 ***
X.PPERSAUT   0.21841    0.02811   7.771 7.81e-15 ***
X.MOPLLAAG  -0.11507    0.04543  -2.533  0.01131 *
X.PBRAND     0.21795    0.07620   2.860  0.00424 **
X.MRELGE     0.18599    0.08371   2.222  0.02629 *
X.MBERBOER  -0.19318    0.08766  -2.204  0.02754 *
X.MINKGEM    0.10409    0.05523   1.885  0.05948 .
X.ABRAND    -0.39044    0.29700  -1.315  0.18865
X.PWAPART    0.56618    0.42226   1.341  0.17998
X.ABROM     -0.39904    0.39142  -1.019  0.30799
X.MRELOV     0.06515    0.09368   0.695  0.48679
X.MSKC       0.06822    0.04965   1.374  0.16942
X.MOPLHOOG   0.05734    0.05055   1.134  0.25663
X.AWAPART   -0.78035    0.84110  -0.928  0.35353
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1912.6  on 4075  degrees of freedom
Residual deviance: 1708.6  on 4062  degrees of freedom
AIC: 1736.6

Number of Fisher Scoring iterations: 6

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

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1600   77
         1   54   15

               Accuracy : 0.925
                 95% CI : (0.9116, 0.9369)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.99997

                  Kappa : 0.1478
 Mcnemar's Test P-Value : 0.05459

            Sensitivity : 0.9674
            Specificity : 0.1630
         Pos Pred Value : 0.9541
         Neg Pred Value : 0.2174
             Prevalence : 0.9473
         Detection Rate : 0.9164
   Detection Prevalence : 0.9605
      Balanced Accuracy : 0.5652

       'Positive' Class : 0
```

We can observe from the summary and the confusion matrix that the accuracy of the model is 92.5% with close to 97% specificity and 16%. we have considered the threshold of 0.2 for positive CARAVAN policy. Any increaseor decrease in the threshold will either hamper specificity or the sensitivity of the model and hence keeping 0.2 as the threshold will be a wise decision.

We can see also see from the summary that there are features which do not contribute to any sort of significance towards the model accuracy. we will get rid of those features and try fitting a model once again.

In [156]:

```
model <- train(as.factor(X.CARAVAN) ~ X.PPERSAUT+X.MOPLLAAG+X.PBRAND+X.MRELGE+X.
MBERBOER+X.MINKGEM,
               data = train,
               trControl = train_control,
               method = "glm",
               family=binomial())

# print cv scores
summary(model)

pred = as.data.frame(predict(model, newdata=test,type="prob"))
pred$new_prob <- ifelse(pred$'1' > 0.2,1,0)
print("------------------------------------------------------------")
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Call:
NULL

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9802  -0.4016  -0.2771  -0.1868   3.1201

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.78717    0.38532 -12.424  < 2e-16 ***
X.PPERSAUT   0.23442    0.02744   8.545  < 2e-16 ***
X.MOPLLAAG  -0.10125    0.03203  -3.161 0.001571 **
X.PBRAND     0.16921    0.03520   4.808 1.53e-06 ***
X.MRELGE     0.13490    0.04046   3.334 0.000856 ***
X.MBERBOER  -0.22874    0.08278  -2.763 0.005722 **
X.MINKGEM    0.11574    0.05183   2.233 0.025541 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1912.6  on 4075  degrees of freedom
Residual deviance: 1719.2  on 4069  degrees of freedom
AIC: 1733.2

Number of Fisher Scoring iterations: 6

[1] "------------------------------------------------------------"
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1611   81
         1   43   11

               Accuracy : 0.929
                 95% CI : (0.9159, 0.9406)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.9995612

                  Kappa : 0.1162
 Mcnemar's Test P-Value : 0.0008915

            Sensitivity : 0.9740
            Specificity : 0.1196
         Pos Pred Value : 0.9521
         Neg Pred Value : 0.2037
             Prevalence : 0.9473
         Detection Rate : 0.9227
   Detection Prevalence : 0.9691
      Balanced Accuracy : 0.5468

       'Positive' Class : 0
```

As expected excluding these features has caused the Sensitivity to cross 97% and the specificity to come down to 11%. Excluding these features has increased the reliability of the model.

6.1.2 Logisitic Regression with Overlapping features

We will now build a model for Overlapping features for the same model.

In [157]:

```
model <- train(as.factor(X.CARAVAN) ~ X.MKOOPKLA+X.PWAPART+X.PPERSAUT+X.PBRAND+
X.ABRAND+X.APERSAUT+X.AWAPART,
               data = train,
               trControl = train_control,
               method = "glm",
               family=binomial())

# print cv scores
summary(model)

pred = as.data.frame(predict(model, newdata=test,type="prob"))
pred$new_prob <- ifelse(pred$'1' > 0.2,1,0)
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Call:
NULL

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.9921  -0.3985  -0.2799  -0.1928   3.0231

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.73289    0.22894 -20.673  < 2e-16 ***
X.MKOOPKLA   0.17379    0.03325   5.227 1.72e-07 ***
X.PWAPART    0.58771    0.40935   1.436  0.15108
X.PPERSAUT   0.17674    0.04681   3.775  0.00016 ***
X.PBRAND     0.16792    0.07251   2.316  0.02057 *
X.ABRAND    -0.28047    0.28367  -0.989  0.32281
X.APERSAUT   0.24164    0.19190   1.259  0.20795
X.AWAPART   -0.74883    0.81493  -0.919  0.35815
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1912.6  on 4075  degrees of freedom
Residual deviance: 1741.6  on 4068  degrees of freedom
AIC: 1757.6

Number of Fisher Scoring iterations: 6
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1630   85
         1   24    7

               Accuracy : 0.9376
                 95% CI : (0.9252, 0.9485)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.967

                  Kappa : 0.0896
 Mcnemar's Test P-Value : 9.086e-09

            Sensitivity : 0.98549
            Specificity : 0.07609
         Pos Pred Value : 0.95044
         Neg Pred Value : 0.22581
             Prevalence : 0.94731
         Detection Rate : 0.93356
   Detection Prevalence : 0.98225
      Balanced Accuracy : 0.53079

       'Positive' Class : 0
```

We can see that the overlapping features have produced better results than the suggested features. The model fares better in terms of specificity and Sensitivity. Let us try some modification with interaction among the realted features. For example the features APERSAUT and PPERSAUT are correlated in the sense the former fearure talks about the a quantity and the latter feature about a measure. the interaction between these two features can lead to better accuracy and performance.

In [160]:

```
model <- train(as.factor(X.CARAVAN) ~ X.MKOOPKLA+X.PWAPART:X.AWAPART+X.PBRAND:X.
ABRAND+X.APERSAUT:X.PPERSAUT,
               data = train,
               trControl = train_control,
               method = "glm",
               family=binomial())

# print cv scores
summary(model)

pred = as.data.frame(predict(model, newdata=test,type="prob"))
pred$new_prob <- ifelse(pred$'1' > 0.2,1,0)
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Call:
NULL

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4299  -0.3856  -0.2965  -0.2311   2.9268

Coefficients:
                       Estimate Std. Error z value Pr(>|z|)
(Intercept)            -4.44938    0.20160 -22.070  < 2e-16 ***
X.MKOOPKLA              0.18033    0.03295   5.473 4.41e-08 ***
`X.PWAPART:X.AWAPART`   0.25264    0.06999   3.610 0.000306 ***
`X.PBRAND:X.ABRAND`     0.05129    0.02814   1.823 0.068341 .
`X.APERSAUT:X.PPERSAUT` 0.13433    0.01618   8.300  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1912.6  on 4075  degrees of freedom
Residual deviance: 1765.6  on 4071  degrees of freedom
AIC: 1775.6

Number of Fisher Scoring iterations: 6
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1625   88
         1   29    4

               Accuracy : 0.933
                 95% CI : (0.9202, 0.9443)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.9959

                  Kappa : 0.0372
 Mcnemar's Test P-Value : 8.226e-08

            Sensitivity : 0.98247
            Specificity : 0.04348
         Pos Pred Value : 0.94863
         Neg Pred Value : 0.12121
             Prevalence : 0.94731
         Detection Rate : 0.93070
   Detection Prevalence : 0.98110
      Balanced Accuracy : 0.51297

       'Positive' Class : 0
```

Surprisingly, It doesnt make a difference in terms of Sensitivity but the percentage of true negative ; Specificity has come down considerably. Therefore, we can consider this a better model than the previous one without interactions. The interaction model presents a better true postive rate and a less true negative rate.

6.2 Naive Bayes Classifier

Let us perform Naive Bayes classification on dataset , which is already split into test and train. We will follow the same procedure overall to arrive at a good model. We will first apply all suggested features and then apply features that are most overlapping in the above analysis

6.2.1 Naive Bayes Classification with Suggested Features

In [196]:

```
naive <- naiveBayes(as.factor(X.CARAVAN)~X.PPERSAUT+X.MOPLLAAG+X.PBRAND+X.MRELGE
+X.MBERBOER+X.MINKGEM+X.ABRAND+X.PWAPART+X.ABROM+X.MRELOV+X.MSKC+X.MOPLHOOG+X.AW
APART,data=train)
naive
pred <- as.data.frame(predict(naive,newdata = test,type="raw"))
pred$new_prob <- ifelse(pred$'0' > 0.1,0,1)
#pred
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
         0          1
0.93719333 0.06280667

Conditional probabilities:
   X.PPERSAUT
Y       [,1]     [,2]
  0 2.852356 2.911847
  1 4.718750 2.423537

   X.MOPLLAAG
Y       [,1]     [,2]
  0 4.592147 2.274128
  1 3.750000 2.294067

   X.PBRAND
Y       [,1]     [,2]
  0 1.774346 1.877464
  1 2.527344 1.816923

   X.MRELGE
Y       [,1]     [,2]
  0 6.123822 1.921096
  1 6.777344 1.592025

   X.MBERBOER
Y        [,1]     [,2]
  0 0.5290576 1.065758
  1 0.3164062 0.744578

   X.MINKGEM
Y       [,1]     [,2]
  0 3.767016 1.324794
  1 4.300781 1.313340

   X.ABRAND
Y        [,1]      [,2]
  0 0.5586387 0.5674843
  1 0.7070312 0.5049843

   X.PWAPART
Y        [,1]      [,2]
  0 0.7502618 0.9523540
  1 1.1367188 0.9905729

   X.ABROM
Y         [,1]      [,2]
  0 0.07513089 0.2743449
  1 0.02734375 0.1634025

   X.MRELOV
Y       [,1]     [,2]
  0 2.339791 1.742297
  1 1.792969 1.471330
```

```
   X.MSKC
Y       [,1]     [,2]
  0 3.743979 1.920085
  1 3.414062 2.023499

   X.MOPLHOOG
Y       [,1]     [,2]
  0 1.447906 1.616444
  1 1.996094 1.800867

   X.AWAPART
Y        [,1]      [,2]
  0 0.3934555 0.4907194
  1 0.5781250 0.4948262
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1630   84
         1   24    8

               Accuracy : 0.9381
                 95% CI : (0.9258, 0.949)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.9588

                  Kappa : 0.1047
 Mcnemar's Test P-Value : 1.369e-08

            Sensitivity : 0.98549
            Specificity : 0.08696
         Pos Pred Value : 0.95099
         Neg Pred Value : 0.25000
             Prevalence : 0.94731
         Detection Rate : 0.93356
   Detection Prevalence : 0.98167
      Balanced Accuracy : 0.53622

       'Positive' Class : 0
```

After a lot of testing with setting a threshold setting a threshold for the probability of the Caravan Policy greater than or equal to 0.1 ie. if the probability of a customer buying a caravan policy is greater than 10% it is more likely to not buy a caravan policy as overall distribition suggests the probability of a customer not buying a caravan policy is more than that of a customer buying one. therefore, setting threshold of 0.1 is totally justified. therefore we can see our model produces close to 99% senstivity and less than 8% Specificity which is really good in terms of accuracy and overall performance

6.2.2 Naive Bayes Classification with Overlapping Features

In [197]:

```
naive <- naiveBayes(as.factor(X.CARAVAN)~X.MKOOPKLA+X.PWAPART+X.PPERSAUT+X.PBRAN
D+X.ABRAND+X.APERSAUT+X.AWAPART,data=train)
naive
pred <- as.data.frame(predict(naive,newdata = test,type="raw"))
pred$new_prob <- ifelse(pred$'0' > 0.1,0,1)
#pred
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
         0          1
0.93719333 0.06280667

Conditional probabilities:
   X.MKOOPKLA
Y       [,1]     [,2]
  0 4.197644 2.009781
  1 5.019531 1.977226

   X.PWAPART
Y        [,1]      [,2]
  0 0.7502618 0.9523540
  1 1.1367188 0.9905729

   X.PPERSAUT
Y       [,1]     [,2]
  0 2.852356 2.911847
  1 4.718750 2.423537

   X.PBRAND
Y       [,1]     [,2]
  0 1.774346 1.877464
  1 2.527344 1.816923

   X.ABRAND
Y        [,1]      [,2]
  0 0.5586387 0.5674843
  1 0.7070312 0.5049843

   X.APERSAUT
Y        [,1]      [,2]
  0 0.5366492 0.5905490
  1 0.9257812 0.5860798

   X.AWAPART
Y        [,1]      [,2]
  0 0.3934555 0.4907194
  1 0.5781250 0.4948262
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1653   92
         1    1    0

               Accuracy : 0.9467
                 95% CI : (0.9351, 0.9568)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.5699

                  Kappa : -0.0011
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9994
            Specificity : 0.0000
         Pos Pred Value : 0.9473
         Neg Pred Value : 0.0000
             Prevalence : 0.9473
         Detection Rate : 0.9467
   Detection Prevalence : 0.9994
      Balanced Accuracy : 0.4997

       'Positive' Class : 0
```

In terms of accuracy and specificity this model perform better than the previous one. However, we can try with interactions among the variables. We can create new columns and include the interaction variables as predictors in the model.

In [198]:

```
train$new_apper <- train$X.APERSAUT*train$X.PPERSAUT
train$new_pwa <- train$X.PWAPART*train$X.AWAPART
train$new_ran <- train$X.ABRAND*train$X.PBRAND
```

In [199]:

```
naive <- naiveBayes(as.factor(X.CARAVAN)~X.MKOOPKLA+new_apper+new_pwa+new_ran,da
ta=train)
naive
pred <- as.data.frame(predict(naive,newdata = test,type="raw"))
pred$new_prob <- ifelse(pred$'0' > 0.1,0,1)
#pred
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
         0          1
0.93719333 0.06280667

Conditional probabilities:
   X.MKOOPKLA
Y       [,1]     [,2]
  0 4.197644 2.009781
  1 5.019531 1.977226

   new_apper
Y       [,1]     [,2]
  0 3.126702 3.560228
  1 5.515625 3.516871

   new_pwa
Y        [,1]      [,2]
  0 0.7534031 0.9646414
  1 1.1367188 0.9905729

   new_ran
Y       [,1]     [,2]
  0 1.905497 2.334486
  1 2.632812 2.049854


Warning message in confusionMatrix.default(data = as.factor(pred$new
_prob), as.factor(test$X.CARAVAN)):
“Levels are not in the same order for reference and data. Refactorin
g data to match.”
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1654   92
         1    0    0

               Accuracy : 0.9473
                 95% CI : (0.9358, 0.9573)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.5277

                  Kappa : 0
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 1.0000
            Specificity : 0.0000
         Pos Pred Value : 0.9473
         Neg Pred Value :    NaN
             Prevalence : 0.9473
         Detection Rate : 0.9473
   Detection Prevalence : 1.0000
      Balanced Accuracy : 0.5000

       'Positive' Class : 0
```

We can see that the sensitivity and specficity show significant improvement. The Naive Bayes is contendors to be the best possible model among the three.

6.3 Linear Discriminant Analysis

Let us perform Linear Discriminant Analysis on dataset , which is already split into test and train. We will follow the same procedure overall to arrive at a good model. We will first apply all suggested features and then apply features that are most overlapping in the above analysis

6.3.1 Discriminant analysis with Suggested Features Features

In [202]:

```
model <- train(as.factor(X.CARAVAN) ~ X.PPERSAUT+X.MOPLLAAG+X.PBRAND+X.MRELGE+X.
MBERBOER+X.MINKGEM+X.ABRAND+X.PWAPART+X.ABROM+X.MRELOV+X.MSKC+X.MOPLHOOG+X.AWAPA
RT,
               data = train,
               trControl = train_control,
               method = "lda")

# print cv scores
summary(model)

pred = as.data.frame(predict(model, newdata=test,type="prob"))
pred$new_prob <- ifelse(pred$'1' > 0.2,1,0)
print("------------------------------------------------------------")
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
            Length Class      Mode
prior        2     -none-     numeric
counts       2     -none-     numeric
means       26     -none-     numeric
scaling     13     -none-     numeric
lev          2     -none-     character
svd          1     -none-     numeric
N            1     -none-     numeric
call         3     -none-     call
xNames      13     -none-     character
problemType  1     -none-     character
tuneValue    1     data.frame list
obsLevels    2     -none-     character
param        0     -none-     list

[1] "------------------------------------------------------------"

Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1598   78
         1   56   14

               Accuracy : 0.9233
                 95% CI : (0.9098, 0.9353)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.99999

                  Kappa : 0.1334
 Mcnemar's Test P-Value : 0.06966

            Sensitivity : 0.9661
            Specificity : 0.1522
         Pos Pred Value : 0.9535
         Neg Pred Value : 0.2000
             Prevalence : 0.9473
         Detection Rate : 0.9152
   Detection Prevalence : 0.9599
      Balanced Accuracy : 0.5592

       'Positive' Class : 0
```

We can observe from the summary statistics and Confusion matrix that the accuracy in terms of sensitivity and specificity is pretty good in comparison to Logisitic regression model. However, we can also try with the overlapping methods and compare model performance.

In [204]:

```
model <- train(as.factor(X.CARAVAN) ~ X.MKOOPKLA+X.PWAPART+X.PPERSAUT+X.PBRAND+
X.ABRAND+X.APERSAUT+X.AWAPART,
               data = train,
               trControl = train_control,
               method = "lda")

# print cv scores
summary(model)

pred = as.data.frame(predict(model, newdata=test,type="prob"))
pred$new_prob <- ifelse(pred$'1' > 0.2,1,0)
confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
            Length Class      Mode
prior        2     -none-     numeric
counts       2     -none-     numeric
means       14     -none-     numeric
scaling      7     -none-     numeric
lev          2     -none-     character
svd          1     -none-     numeric
N            1     -none-     numeric
call         3     -none-     call
xNames       7     -none-     character
problemType  1     -none-     character
tuneValue    1     data.frame list
obsLevels    2     -none-     character
param        0     -none-     list
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1609   83
         1   45    9

               Accuracy : 0.9267
                 95% CI : (0.9134, 0.9385)
    No Information Rate : 0.9473
    P-Value [Acc > NIR] : 0.999898

                  Kappa : 0.0877
 Mcnemar's Test P-Value : 0.001074

            Sensitivity : 0.97279
            Specificity : 0.09783
         Pos Pred Value : 0.95095
         Neg Pred Value : 0.16667
             Prevalence : 0.94731
         Detection Rate : 0.92153
   Detection Prevalence : 0.96907
      Balanced Accuracy : 0.53531

       'Positive' Class : 0
```

we can see from the summary statistics the model performs better than the previous model with suggested features. In conclusion, we can say that in all the three models and the overlapping features have fared better than the suggested Features. We can therefore confirm our hypthesis at the initial stages of this solution regarding the suitability of overlapping variables for this particular problem.

*Finally, considering the Sensitivity and Specificity for all the models, The **Naive Bayes** Classifier shows better performance in comparison to other models. Therefore, we will go ahead predict the proabilities for the test data and compare the model performance.*

---

7. Prediction of Probabilities for Test Data

First let us create the interaction variables. We can confirm from the above cross validation section that Naive Bayes Classifier Works the best with Interaction variables which included the cross product of several other features. Let us go ahead and do that for Train Data

In [205]:

```
train_data$new_apper <- train_data$X.APERSAUT*train_data$X.PPERSAUT
train_data$new_pwa <- train_data$X.PWAPART*train_data$X.AWAPART
train_data$new_ran <- train_data$X.ABRAND*train_data$X.PBRAND
```

In [208]:

```
naive_final <- naiveBayes(as.factor(X.CARAVAN)~X.MKOOPKLA+new_apper+new_pwa+new_
ran,data=train_data)
naive_final
pred <- as.data.frame(predict(naive_final,newdata = test_data,type="raw"))

nrow(pred)
pred$index <- seq.int(nrow(pred))
pred

#pred$new_prob <- ifelse(pred$'0' > 0.1,0,1)
#pred
#confusionMatrix(data=as.factor(pred$new_prob), as.factor(test$X.CARAVAN))
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
         0          1
0.94022673 0.05977327

Conditional probabilities:
   X.MKOOPKLA
Y       [,1]     [,2]
  0 4.187797 1.997301
  1 5.000000 2.010061

   new_apper
Y       [,1]     [,2]
  0 3.153087 3.652594
  1 5.408046 3.393526

   new_pwa
Y        [,1]      [,2]
  0 0.7510047 0.9635813
  1 1.1379310 0.9889582

   new_ran
Y       [,1]     [,2]
  0 1.913592 2.309986
  1 2.623563 2.016993

4000
```

0	1	index
0.9560645	0.04393547	1
0.8863685	0.11363153	2
0.9357910	0.06420900	3
0.9469124	0.05308764	4
0.9697713	0.03022873	5
0.9635882	0.03641178	6
0.9357910	0.06420900	7
0.8863685	0.11363153	8
0.9223034	0.07769663	9
0.8863685	0.11363153	10
0.9223034	0.07769663	11
0.9560645	0.04393547	12
0.9469124	0.05308764	13
0.9223034	0.07769663	14
0.9560645	0.04393547	15
0.9697713	0.03022873	16
0.9469124	0.05308764	17
0.9357910	0.06420900	18
0.9560645	0.04393547	19
0.9635882	0.03641178	20
0.9560645	0.04393547	21
0.9223034	0.07769663	22
0.9560645	0.04393547	23
0.9357910	0.06420900	24
0.9697713	0.03022873	25
0.9635882	0.03641178	26
0.9059965	0.09400350	27
0.9223034	0.07769663	28
0.9560645	0.04393547	29
0.8863685	0.11363153	30
⋮	⋮	⋮
0.9635882	0.03641178	3971
0.8863685	0.11363153	3972
0.9223034	0.07769663	3973

0	1	index
0.8863685	0.11363153	3974
0.9357910	0.06420900	3975
0.9560645	0.04393547	3976
0.9560645	0.04393547	3977
0.9560645	0.04393547	3978
0.9469124	0.05308764	3979
0.9357910	0.06420900	3980
0.9560645	0.04393547	3981
0.9469124	0.05308764	3982
0.8863685	0.11363153	3983
0.9223034	0.07769663	3984
0.9560645	0.04393547	3985
0.9223034	0.07769663	3986
0.9059965	0.09400350	3987
0.9469124	0.05308764	3988
0.9560645	0.04393547	3989
0.9059965	0.09400350	3990
0.9560645	0.04393547	3991
0.9635882	0.03641178	3992
0.9635882	0.03641178	3993
0.9560645	0.04393547	3994
0.8863685	0.11363153	3995
0.9560645	0.04393547	3996
0.9635882	0.03641178	3997
0.9560645	0.04393547	3998
0.9560645	0.04393547	3999
0.8863685	0.11363153	4000

Now that we have learnt a model using the selected features using the entire training dataset and predicted the probabilities on the test set. Now we have to reorder the probabilities in descending order and find the 800 customers who are likely to opt for a caravan policy

In [210]:

```
sort.prob <- pred[order(pred$'1',decreasing = TRUE),]
sort.prob
```

	0	1	index
2	0.8863685	0.1136315	2
8	0.8863685	0.1136315	8
10	0.8863685	0.1136315	10
30	0.8863685	0.1136315	30
58	0.8863685	0.1136315	58
73	0.8863685	0.1136315	73
82	0.8863685	0.1136315	82
89	0.8863685	0.1136315	89
94	0.8863685	0.1136315	94
107	0.8863685	0.1136315	107
110	0.8863685	0.1136315	110
113	0.8863685	0.1136315	113
139	0.8863685	0.1136315	139
148	0.8863685	0.1136315	148
155	0.8863685	0.1136315	155
158	0.8863685	0.1136315	158
183	0.8863685	0.1136315	183
210	0.8863685	0.1136315	210
265	0.8863685	0.1136315	265
269	0.8863685	0.1136315	269
289	0.8863685	0.1136315	289
291	0.8863685	0.1136315	291
296	0.8863685	0.1136315	296
297	0.8863685	0.1136315	297
314	0.8863685	0.1136315	314
317	0.8863685	0.1136315	317
323	0.8863685	0.1136315	323
331	0.8863685	0.1136315	331
332	0.8863685	0.1136315	332
337	0.8863685	0.1136315	337
⋮	⋮	⋮	⋮
3665	0.9697713	0.03022873	3665
3670	0.9697713	0.03022873	3670
3674	0.9697713	0.03022873	3674

	0	1	index
3687	0.9697713	0.03022873	3687
3693	0.9697713	0.03022873	3693
3702	0.9697713	0.03022873	3702
3704	0.9697713	0.03022873	3704
3721	0.9697713	0.03022873	3721
3726	0.9697713	0.03022873	3726
3742	0.9697713	0.03022873	3742
3750	0.9697713	0.03022873	3750
3760	0.9697713	0.03022873	3760
3786	0.9697713	0.03022873	3786
3791	0.9697713	0.03022873	3791
3797	0.9697713	0.03022873	3797
3804	0.9697713	0.03022873	3804
3814	0.9697713	0.03022873	3814
3818	0.9697713	0.03022873	3818
3833	0.9697713	0.03022873	3833
3858	0.9697713	0.03022873	3858
3866	0.9697713	0.03022873	3866
3880	0.9697713	0.03022873	3880
3882	0.9697713	0.03022873	3882
3908	0.9697713	0.03022873	3908
3913	0.9697713	0.03022873	3913
3924	0.9697713	0.03022873	3924
3933	0.9697713	0.03022873	3933
3948	0.9697713	0.03022873	3948
3953	0.9697713	0.03022873	3953
3967	0.9697713	0.03022873	3967

Now that we have sorted the proabilities, let us know reorder the indices and compare the prediction results using a confusion matrix. First let us assign true and False against each of the probabilities

In [211]:

```
sort.prob$new_prob[1:800] <- 1
sort.prob$new_prob[801:nrow(sort.prob)] <- 0
sort.prob
```

	0	1	index	new_prob
2	0.8863685	0.1136315	2	1
8	0.8863685	0.1136315	8	1
10	0.8863685	0.1136315	10	1
30	0.8863685	0.1136315	30	1
58	0.8863685	0.1136315	58	1
73	0.8863685	0.1136315	73	1
82	0.8863685	0.1136315	82	1
89	0.8863685	0.1136315	89	1
94	0.8863685	0.1136315	94	1
107	0.8863685	0.1136315	107	1
110	0.8863685	0.1136315	110	1
113	0.8863685	0.1136315	113	1
139	0.8863685	0.1136315	139	1
148	0.8863685	0.1136315	148	1
155	0.8863685	0.1136315	155	1
158	0.8863685	0.1136315	158	1
183	0.8863685	0.1136315	183	1
210	0.8863685	0.1136315	210	1
265	0.8863685	0.1136315	265	1
269	0.8863685	0.1136315	269	1
289	0.8863685	0.1136315	289	1
291	0.8863685	0.1136315	291	1
296	0.8863685	0.1136315	296	1
297	0.8863685	0.1136315	297	1
314	0.8863685	0.1136315	314	1
317	0.8863685	0.1136315	317	1
323	0.8863685	0.1136315	323	1
331	0.8863685	0.1136315	331	1
332	0.8863685	0.1136315	332	1
337	0.8863685	0.1136315	337	1
⋮	⋮	⋮	⋮	⋮
3665	0.9697713	0.03022873	3665	0
3670	0.9697713	0.03022873	3670	0
3674	0.9697713	0.03022873	3674	0

	0	1	index	new_prob
3687	0.9697713	0.03022873	3687	0
3693	0.9697713	0.03022873	3693	0
3702	0.9697713	0.03022873	3702	0
3704	0.9697713	0.03022873	3704	0
3721	0.9697713	0.03022873	3721	0
3726	0.9697713	0.03022873	3726	0
3742	0.9697713	0.03022873	3742	0
3750	0.9697713	0.03022873	3750	0
3760	0.9697713	0.03022873	3760	0
3786	0.9697713	0.03022873	3786	0
3791	0.9697713	0.03022873	3791	0
3797	0.9697713	0.03022873	3797	0
3804	0.9697713	0.03022873	3804	0
3814	0.9697713	0.03022873	3814	0
3818	0.9697713	0.03022873	3818	0
3833	0.9697713	0.03022873	3833	0
3858	0.9697713	0.03022873	3858	0
3866	0.9697713	0.03022873	3866	0
3880	0.9697713	0.03022873	3880	0
3882	0.9697713	0.03022873	3882	0
3908	0.9697713	0.03022873	3908	0
3913	0.9697713	0.03022873	3913	0
3924	0.9697713	0.03022873	3924	0
3933	0.9697713	0.03022873	3933	0
3948	0.9697713	0.03022873	3948	0
3953	0.9697713	0.03022873	3953	0
3967	0.9697713	0.03022873	3967	0

Let us reorder the indices to get the results in the correct order.

In [212]:

```
sort.prob <- sort.prob[order(sort.prob$index),]
sort.prob
```

0	1	index	new_prob
0.9560645	0.04393547	1	0
0.8863685	0.11363153	2	1
0.9357910	0.06420900	3	0
0.9469124	0.05308764	4	0
0.9697713	0.03022873	5	0
0.9635882	0.03641178	6	0
0.9357910	0.06420900	7	0
0.8863685	0.11363153	8	1
0.9223034	0.07769663	9	1
0.8863685	0.11363153	10	1
0.9223034	0.07769663	11	1
0.9560645	0.04393547	12	0
0.9469124	0.05308764	13	0
0.9223034	0.07769663	14	1
0.9560645	0.04393547	15	0
0.9697713	0.03022873	16	0
0.9469124	0.05308764	17	0
0.9357910	0.06420900	18	0
0.9560645	0.04393547	19	0
0.9635882	0.03641178	20	0
0.9560645	0.04393547	21	0
0.9223034	0.07769663	22	1
0.9560645	0.04393547	23	0
0.9357910	0.06420900	24	0
0.9697713	0.03022873	25	0
0.9635882	0.03641178	26	0
0.9059965	0.09400350	27	1
0.9223034	0.07769663	28	1
0.9560645	0.04393547	29	0
0.8863685	0.11363153	30	1
⋮	⋮	⋮	⋮
0.9635882	0.03641178	3971	0
0.8863685	0.11363153	3972	1
0.9223034	0.07769663	3973	0

0	1	index	new_prob
0.8863685	0.11363153	3974	1
0.9357910	0.06420900	3975	0
0.9560645	0.04393547	3976	0
0.9560645	0.04393547	3977	0
0.9560645	0.04393547	3978	0
0.9469124	0.05308764	3979	0
0.9357910	0.06420900	3980	0
0.9560645	0.04393547	3981	0
0.9469124	0.05308764	3982	0
0.8863685	0.11363153	3983	1
0.9223034	0.07769663	3984	0
0.9560645	0.04393547	3985	0
0.9223034	0.07769663	3986	0
0.9059965	0.09400350	3987	1
0.9469124	0.05308764	3988	0
0.9560645	0.04393547	3989	0
0.9059965	0.09400350	3990	1
0.9560645	0.04393547	3991	0
0.9635882	0.03641178	3992	0
0.9635882	0.03641178	3993	0
0.9560645	0.04393547	3994	0
0.8863685	0.11363153	3995	1
0.9560645	0.04393547	3996	0
0.9635882	0.03641178	3997	0
0.9560645	0.04393547	3998	0
0.9560645	0.04393547	3999	0
0.8863685	0.11363153	4000	1

Now let us ascertain the performance of the Naive Bayes Classifier using the eval dataset and describe the factors leading to the predictions and come up with insights to the marketing team

In [213]:

```
confusionMatrix(data=as.factor(sort.prob$new_prob), as.factor(eval_data$V1))
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 3044  156
         1  718   82

               Accuracy : 0.7815
                 95% CI : (0.7684, 0.7942)
    No Information Rate : 0.9405
    P-Value [Acc > NIR] : 1

                  Kappa : 0.073
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.8091
            Specificity : 0.3445
         Pos Pred Value : 0.9512
         Neg Pred Value : 0.1025
             Prevalence : 0.9405
         Detection Rate : 0.7610
   Detection Prevalence : 0.8000
      Balanced Accuracy : 0.5768

       'Positive' Class : 0
```

Overall Accuracy has come down compared to the other model. However, the Sensitivity shows a good result and indicates that the model was successfull in identify 81% true positive value. We will use these results to form the basis of further description and insights

---

8. Why a Customer Buys Caravan Policy ???

From covering the entire Methodology and understanding how a model is built, to disocovering ways to get rid of "Curse of Dimensionality" we have understood the overall strucure of the methodology and the comprehensive approach that was employed. Throughout the assessment we realised how the significance of features contributes to the significance of the results. In order to arrive at a plausible explanation of what lead to a particular customer being tagged as "NO" or "YES" we will have to unravel the features and understand the contribution of these features towards the final result. This in a sense is like "Reverse Engineering" where we go back from the results to the causes that lead to the result. The Question "Why Customer Buys Caravan Policy" is extremely significant from the insurance company's point of view. The need is to find the sections/attributes in a particular customer base and target those sections by employing severl ways : which could include : Investment, enhancing infrastructure, Digitization or perhaps advertisment. In this section we will only look at the TRUE POSITIVE predictions (Sensitivity) and draw insights and conclusions from it.

In [216]:

```
test_data$CARAVAN <- sort.prob$new_prob
test_data$eval_data <- eval_data$V1
```

In [217]:

```
table(test_data$CARAVAN,test_data$eval_data)
```

```
      0    1
  0 3044  156
  1  718   82
```

In [236]:

```
true_pos_data <- test_data[((test_data$CARAVAN == 0) & (test_data$eval_data == 0
)) | ((test_data$CARAVAN == 1) & (test_data$eval_data == 1)),]
```

In [237]:

```
nrow(true_pos_data)
ncol(true_pos_data)
```

3126

87

We have now gathered all the data which has TRUE POSITIVE results. Now let us extract only the features which were used to predict the proabaility outcomes

In [238]:

```
head(true_pos_data)
```

X.MOSTYPE	X.MAANTHUI	X.MGEMOMV	X.MGEMLEEF	X.MOSHOOFD	X.MGODRK
33	1	4	2	8	0
6	1	3	2	2	0
39	1	3	3	9	1
9	1	2	3	3	2
31	1	2	4	7	0
30	1	2	4	7	1

In [241]:

```
true_pos_data <- true_pos_data[,c('X.MKOOPKLA','X.PWAPART','X.PPERSAUT','X.PBRAN
D','X.ABRAND','X.APERSAUT','X.AWAPART','CARAVAN')]
```

In [243]:

```
head(true_pos_data)
```

X.MKOOPKLA	X.PWAPART	X.PPERSAUT	X.PBRAND	X.ABRAND	X.APERSAUT	X.A
3	1	0	4	1	0	1
8	2	6	4	1	1	1
5	2	6	4	1	1	1
4	2	5	3	1	1	1
1	2	0	1	1	0	1
2	0	0	4	2	0	0

Now that we subsetted the True Postivie data, let us subset again and obtain the dataset wherein the customer actually bought a caravan policy and the predicted value was true too.

In [244]:

```
true_pos_data <- true_pos_data[true_pos_data$CARAVAN == 1,]
nrow(true_pos_data)
```

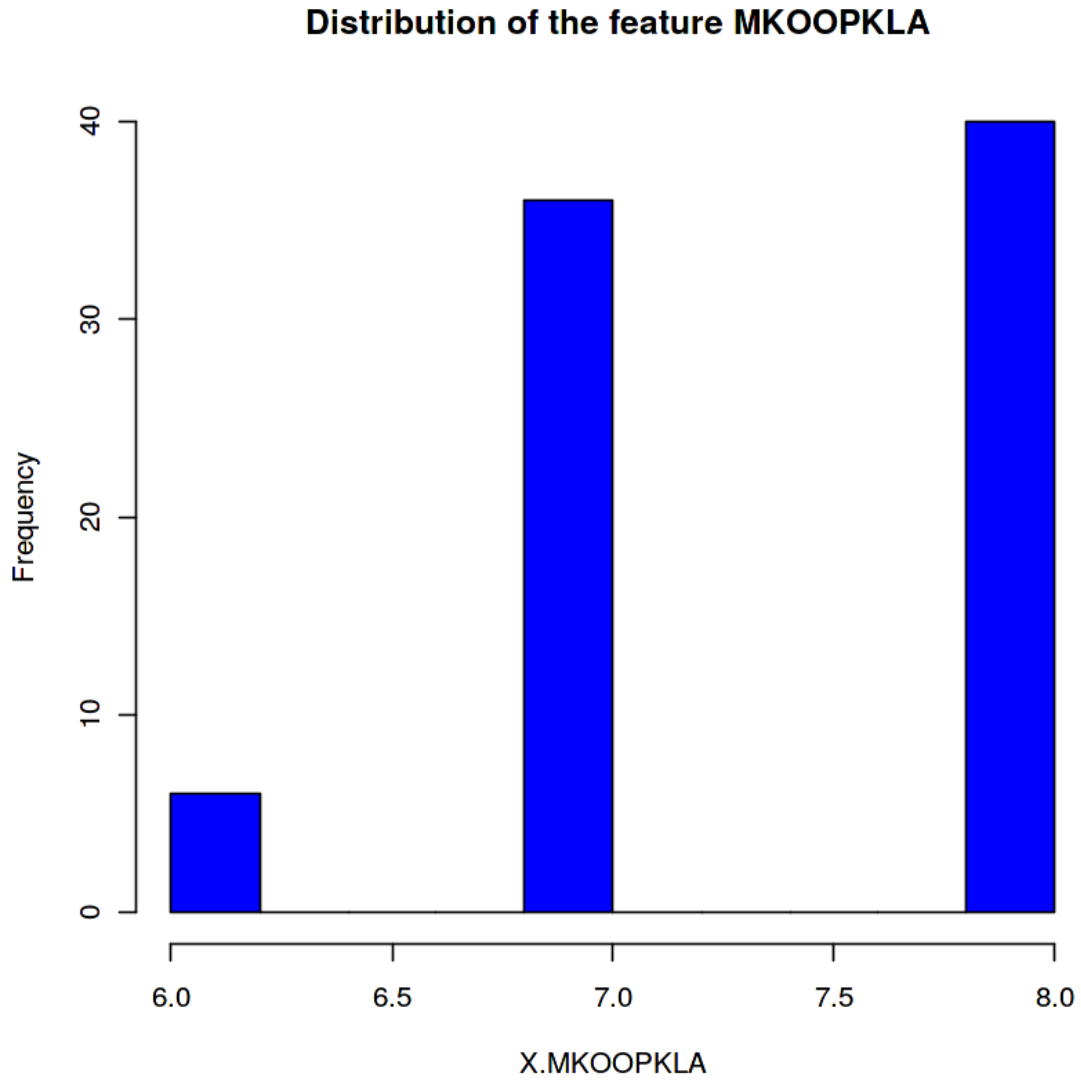
82

Now that we have subsetted only those rows wherein the customer actually buys a caravan policy and the same was predicted , let us examine the distribution of each of the features.

8.1 Exploring The Purchaing Power Class(MKOOPKLA)

In [271]:

```
hist(true_pos_data[, as.factor('X.MKOOPKLA')],xlab = 'X.MKOOPKLA', main=paste("
 Distribution of the feature MKOOPKLA"),col="blue")
```



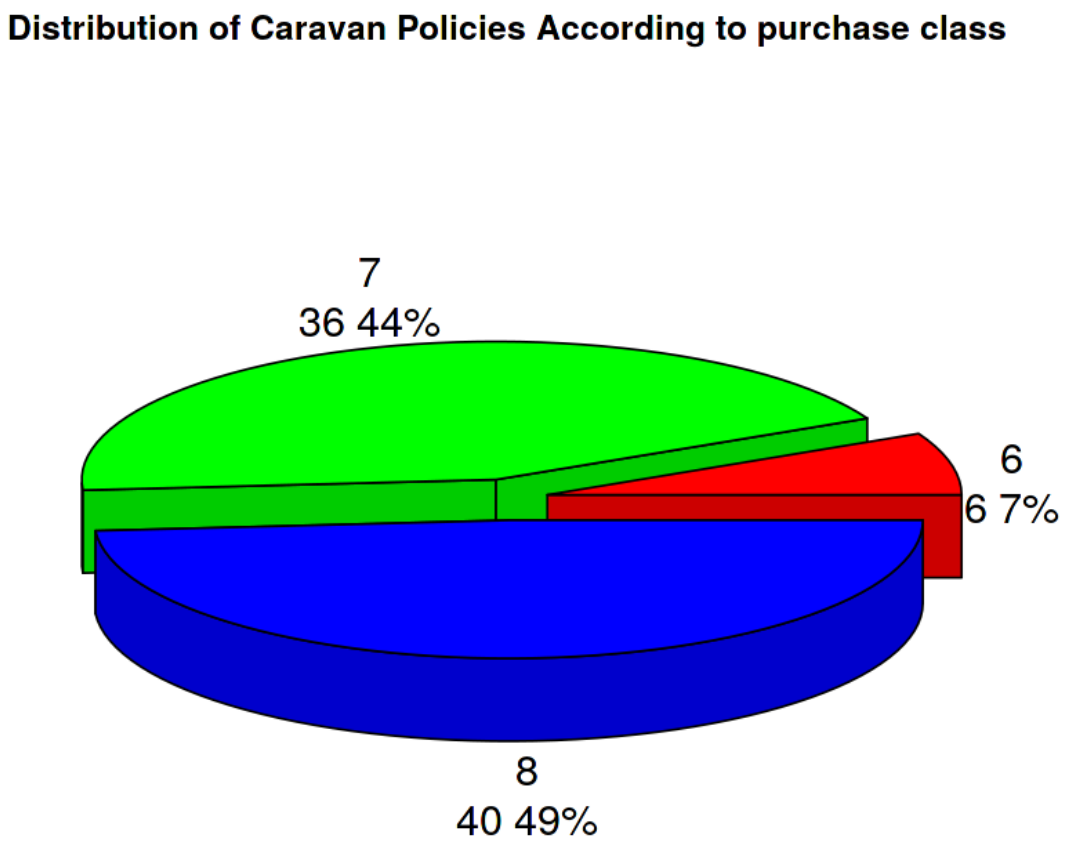
In [272]:

```
dist_MKOOPKLA <- table(true_pos_data$X.MKOOPKLA)
dist_MKOOPKLA
```

```
 6  7  8
 6 36 40
```

In [274]:

```
lbls <- paste(names(dist_MKOOPKLA), "\n", dist_MKOOPKLA, sep="")
pct <- round(dist_MKOOPKLA/sum(dist_MKOOPKLA)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(dist_MKOOPKLA,labels=lbls,explode=0.1,
   main="Distribution of Caravan Policies According to purchase class")
```

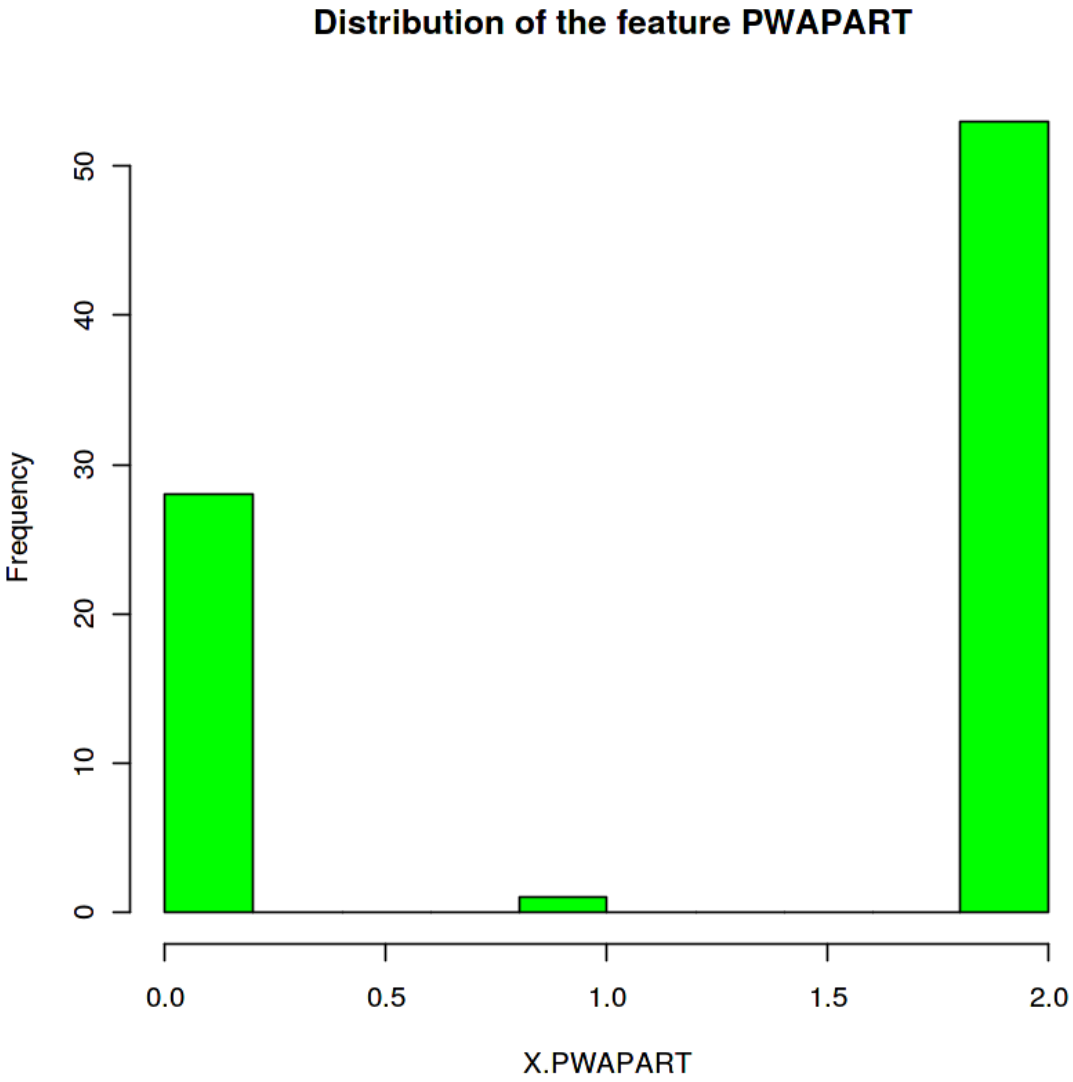


*We can observe from the histogram and more clearly from the pie chart, that Purchasing Power class is perhaps the single most important feature when it comes deriving predictability using demographic data. We can see that **49%** of the customers who bought CARAVAn policy come from subclass 8, closely followed by class 7 with 44% and class 6 with 7%. Therefore, the company has to target more customers with subclass 7 or 8 in order to inrease the customer base*

8.2 Exploring Contribution to Third Party Insurance

In [284]:

```
hist(true_pos_data[,'X.PWAPART'],xlab = 'X.PWAPART', main=paste(" Distribution o
f the feature PWAPART"),col="green")
```



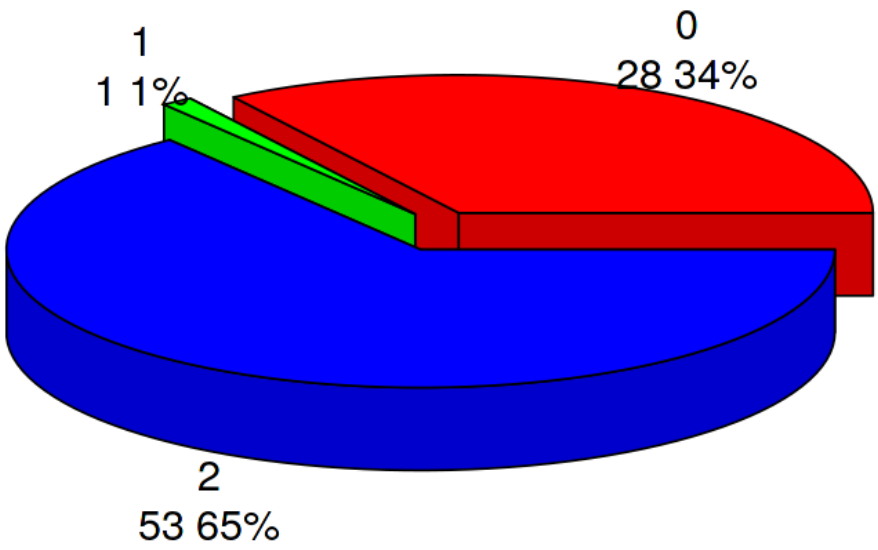
In [278]:

```
dist_PWAPART <- table(true_pos_data$X.PWAPART)
dist_PWAPART
```

```
 0  1  2
28  1 53
```

In [287]:

```
lbls <- paste(names(dist_PWAPART), "\n", dist_PWAPART, sep="")
pct <- round(dist_PWAPART/sum(dist_PWAPART)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(dist_PWAPART,labels=lbls,explode=0.1,
   main="Distribution of Caravan Policies wrt to PWAPART")
```

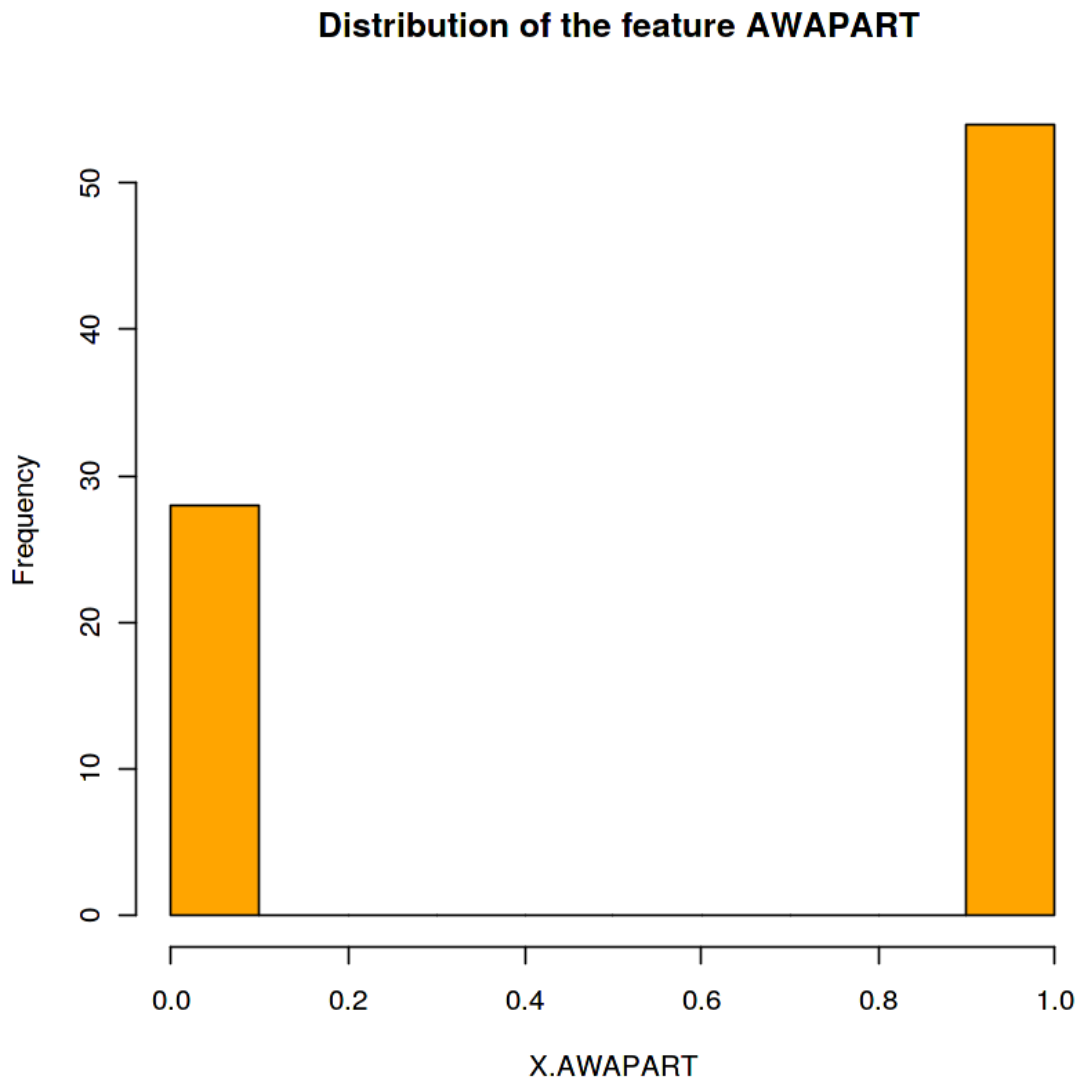


We can see that customers with either 0 contributions or 2 third party insurance policies tend to buy caravan policies. The Possible reason could be , with customers who did not buy any policies want to have one and the customers with 2 contributions want to safeguard their contributions already. Therefore, the easier bet is to target customers with no or 2 contributions to expect an opt-in for Caravan Policy

8.3 Exploring Number of Third Party Insurance

In [291]:

```
hist(true_pos_data[,'X.AWAPART'],xlab = 'X.AWAPART', main=paste(" Distribution o
f the feature AWAPART"),col="orange")
```

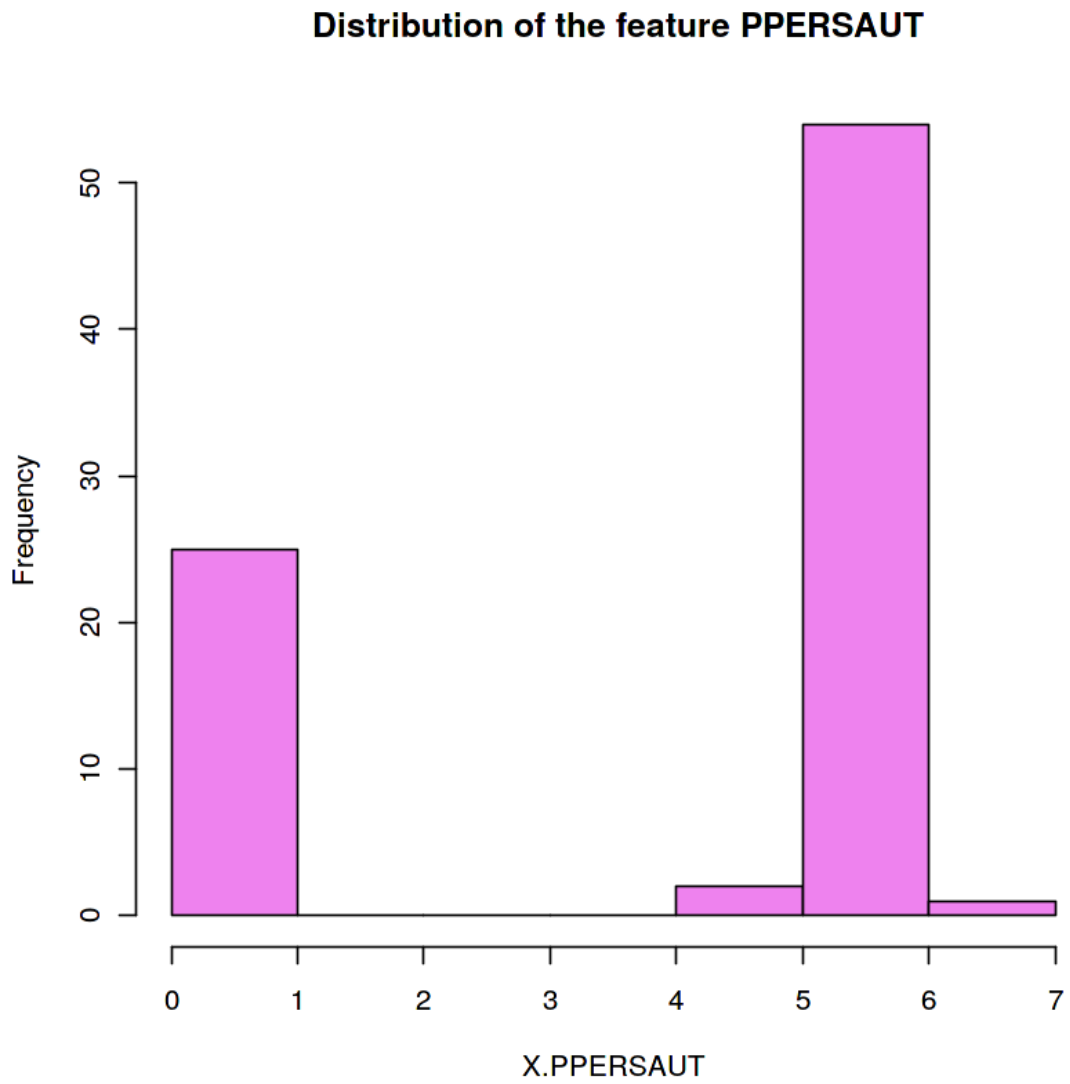


We can observe from the distribution that customers with 1 third party insurance are more likely to opt for a caravan policy insurance.

8.4 Exploring Contribution to car policies

In [292]:

```
hist(true_pos_data[,'X.PPERSAUT'],xlab = 'X.PPERSAUT', main=paste(" Distribution
of the feature PPERSAUT"),col="violet")
```



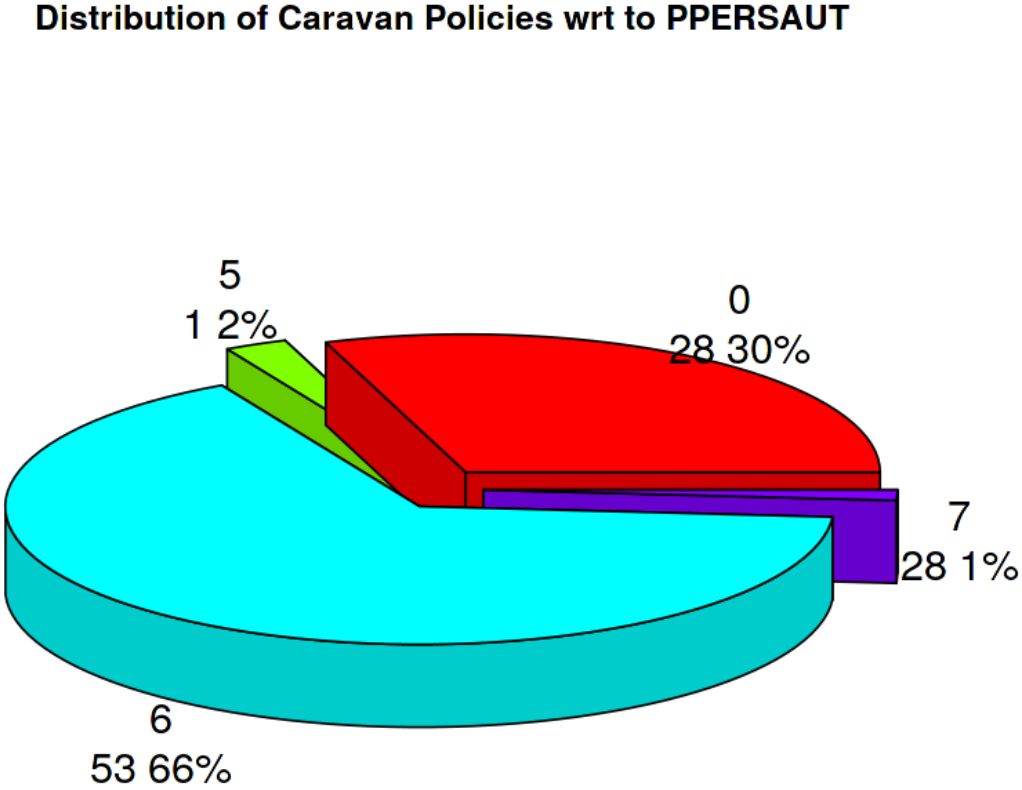
In [294]:

```
dist_PPERSAUT <- table(true_pos_data$X.PPERSAUT)
dist_PPERSAUT
```

```
 0  5  6  7
25  2 54  1
```

In [295]:

```
lbls <- paste(names(dist_PPERSAUT), "\n", dist_PWAPART, sep="")
pct <- round(dist_PPERSAUT/sum(dist_PPERSAUT)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(dist_PPERSAUT,labels=lbls,explode=0.1,
   main="Distribution of Caravan Policies wrt to PPERSAUT")
```

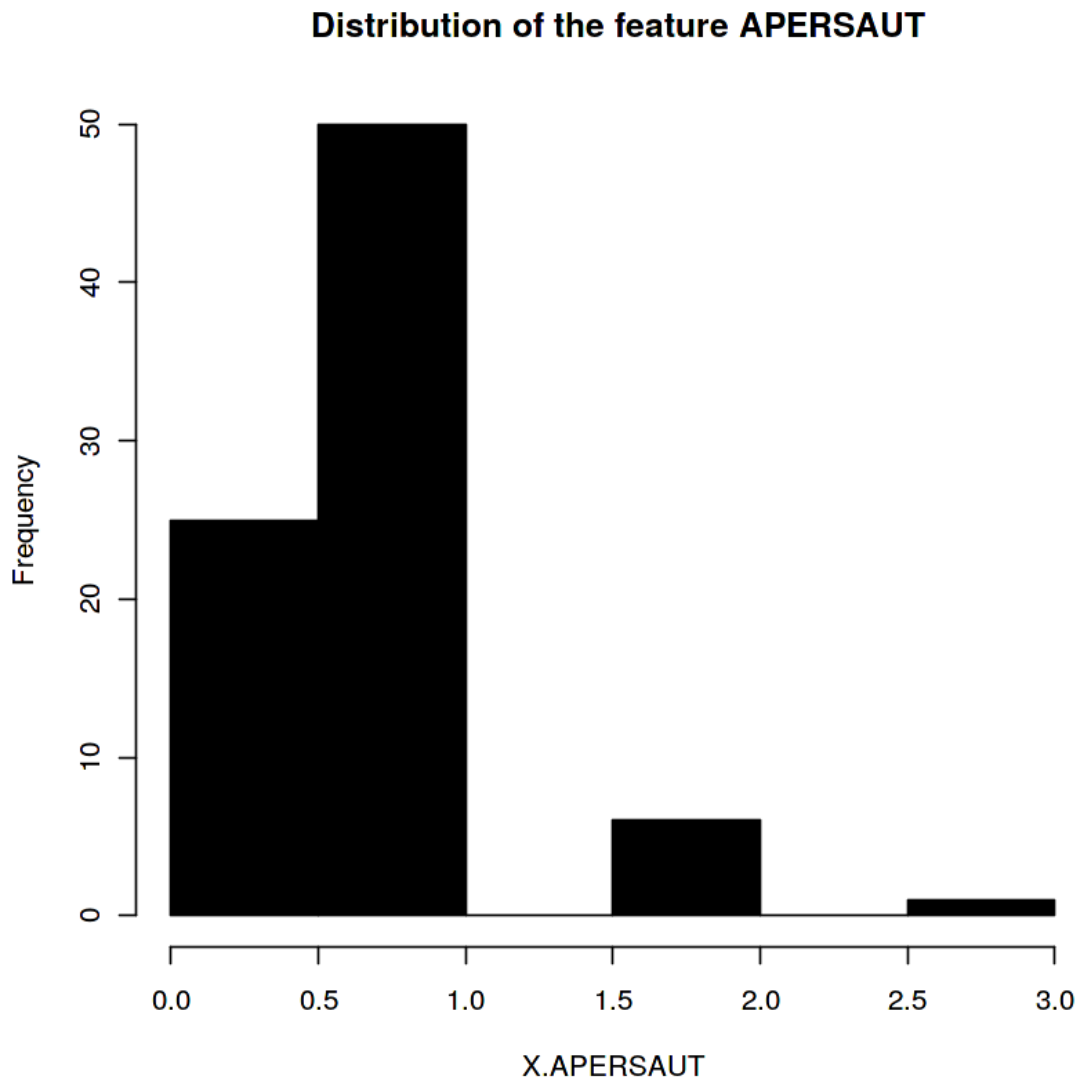


We can observe that 66% of customers with 6 car contribution policies have bought the caravan policy , closely followed by 30% customers who do not have any contribution towards car policies. Therefore, the marketing company should target more the customers with 6 contributions.

8.5 Exploring Number of car policies(APERSAUT)

In [296]:

```
hist(true_pos_data[,'X.APERSAUT'],xlab = 'X.APERSAUT', main=paste(" Distribution
of the feature APERSAUT"),col="black")
```



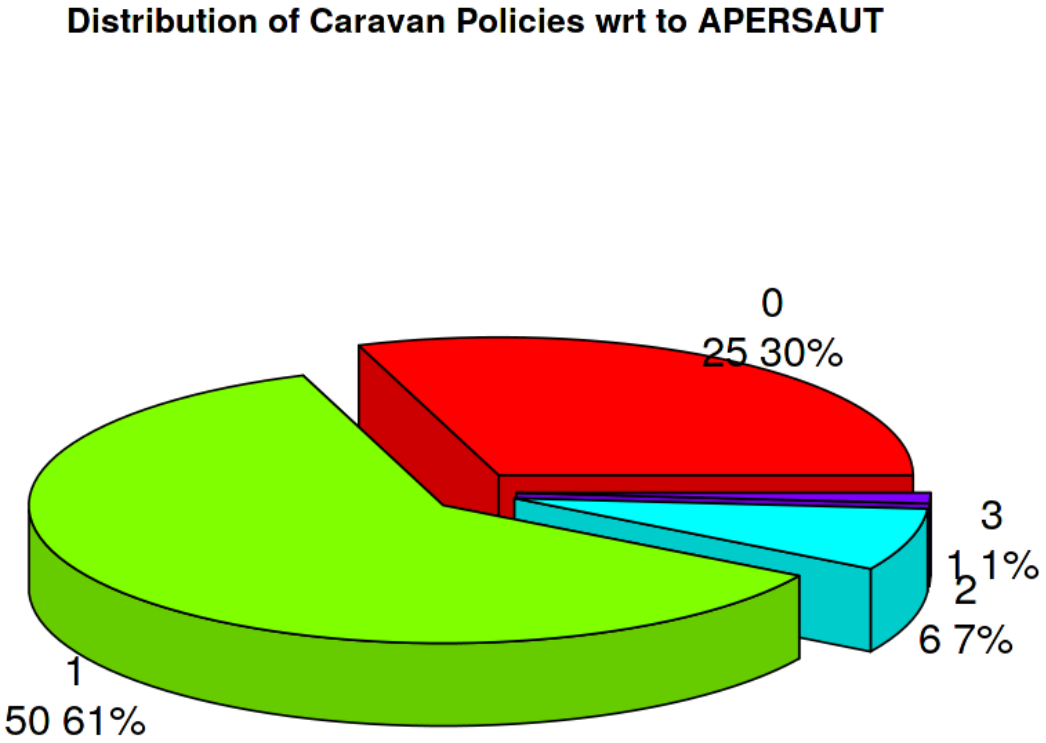
In [297]:

```
dist_APERSAUT <- table(true_pos_data$X.APERSAUT)
dist_APERSAUT
```

```
 0  1  2  3
25 50  6  1
```

In [298]:

```
lbls <- paste(names(dist_APERSAUT), "\n", dist_APERSAUT, sep="")
pct <- round(dist_APERSAUT/sum(dist_APERSAUT)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(dist_APERSAUT,labels=lbls,explode=0.1,
   main="Distribution of Caravan Policies wrt to APERSAUT")
```

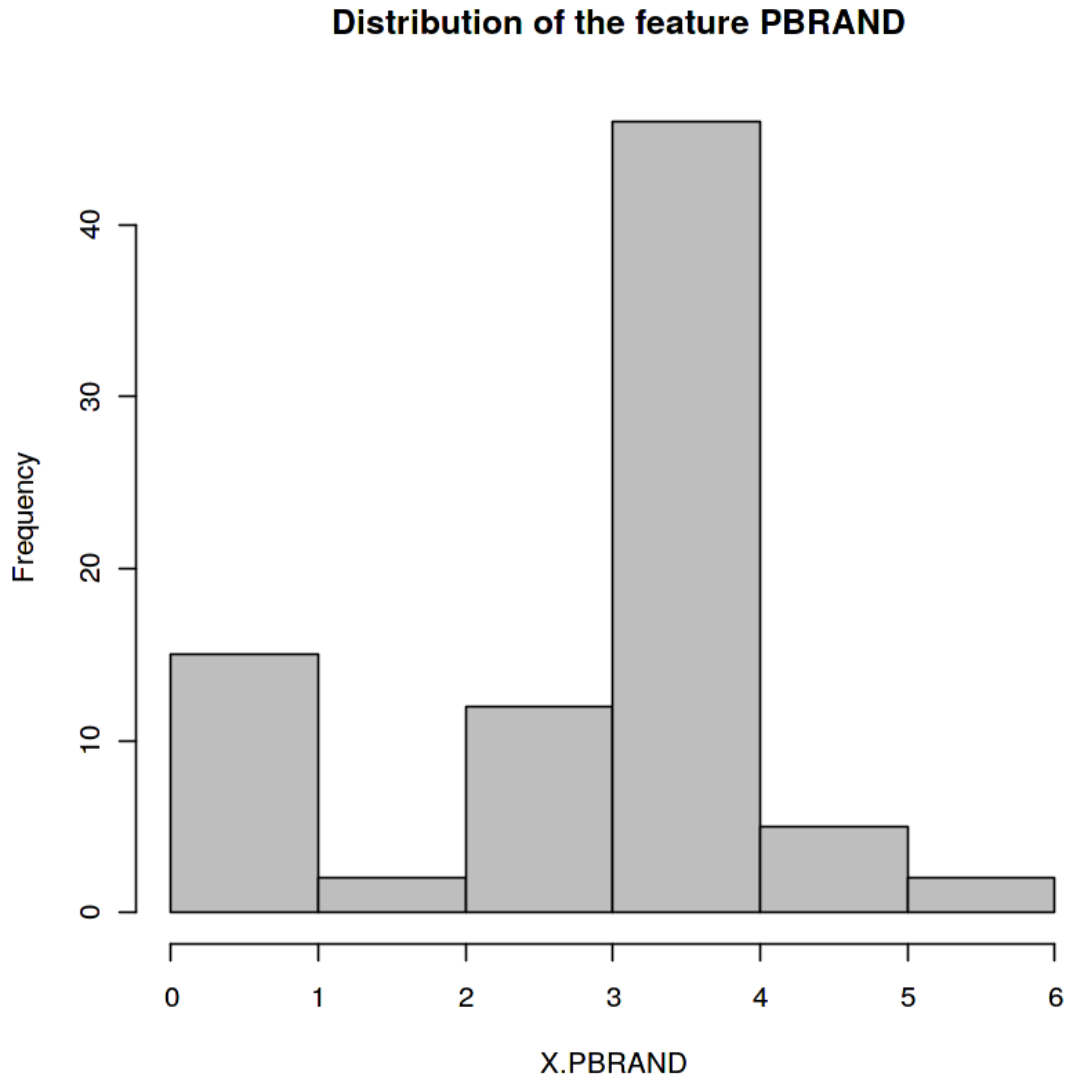


we can observe from the pie chart that customers with 1 car policies, 61% of them have bought the caravan insurance policies. closely followed by customers with 0 car policies. Therefore, the marketing team must target customers with 1 car policy.

8.6 Exploring Contribution to fire policies(PBRAND)

In [299]:

```
hist(true_pos_data[,'X.PBRAND'],xlab = 'X.PBRAND', main=paste(" Distribution of
 the feature PBRAND"),col="gray")
```



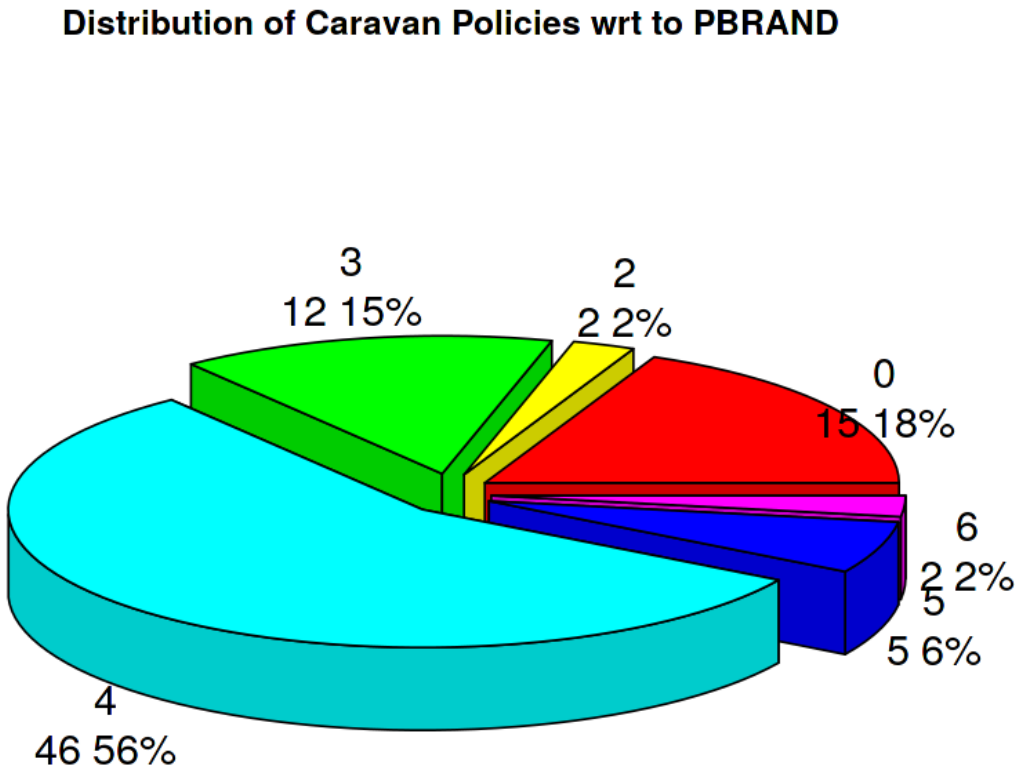
In [303]:

```
dist_PBRAND <- table(true_pos_data$X.PBRAND)
dist_PBRAND
```

```
 0  2  3  4  5  6
15  2 12 46  5  2
```

In [301]:

```
lbls <- paste(names(dist_PBRAND), "\n", dist_PBRAND, sep="")
pct <- round(dist_PBRAND/sum(dist_PBRAND)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(dist_PBRAND,labels=lbls,explode=0.1,
   main="Distribution of Caravan Policies wrt to PBRAND")
```

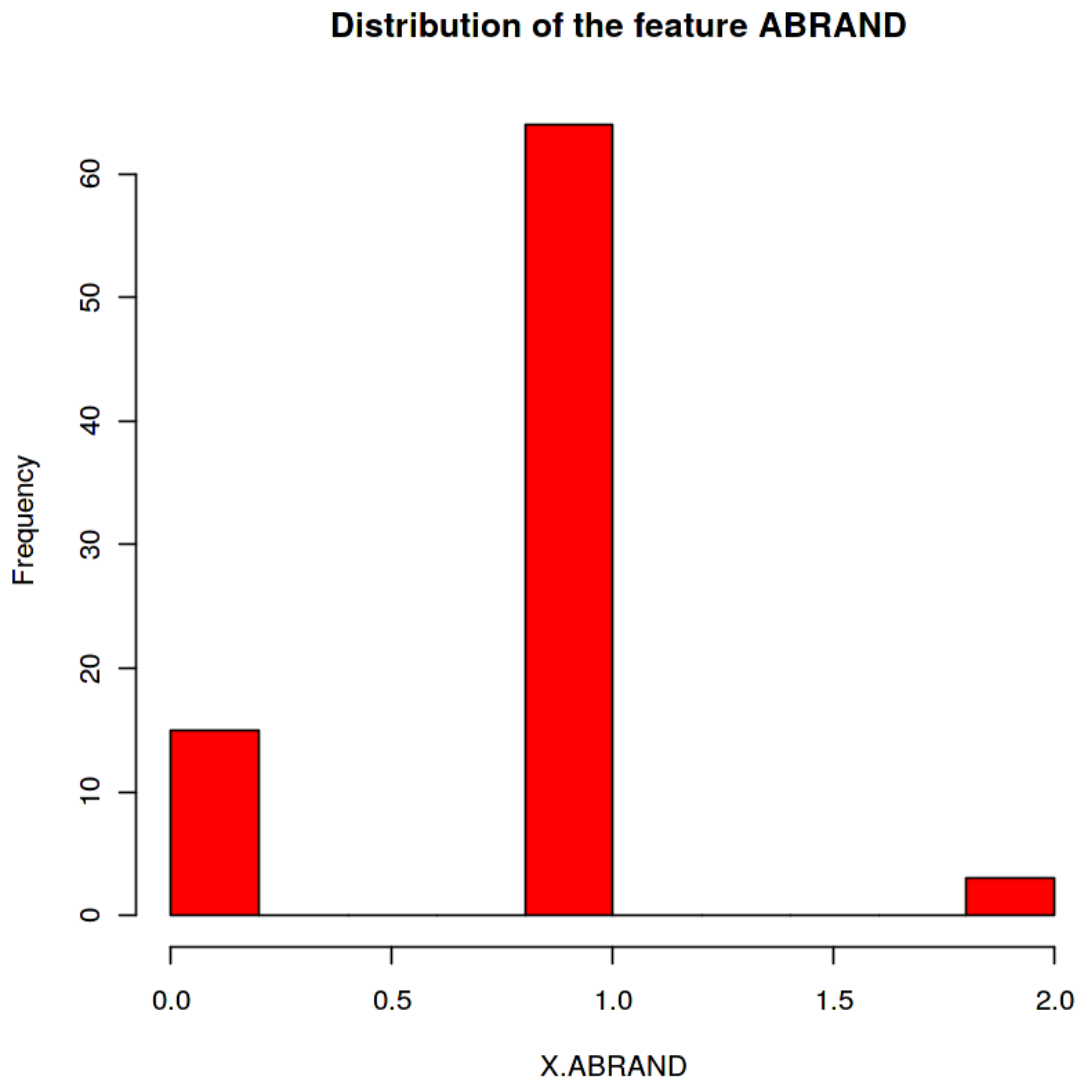


We can see from the pie chart that 56% of the people who bought caravan policy have bought caravan policy . Therefore, the customers with 4 contribution policies must be targeted as well.

8.7 Exploring Number of fire policies(ABRAND)

In [302]:

```
hist(true_pos_data[,'X.ABRAND'],xlab = 'X.ABRAND', main=paste(" Distribution of
 the feature ABRAND"),col="red")
```



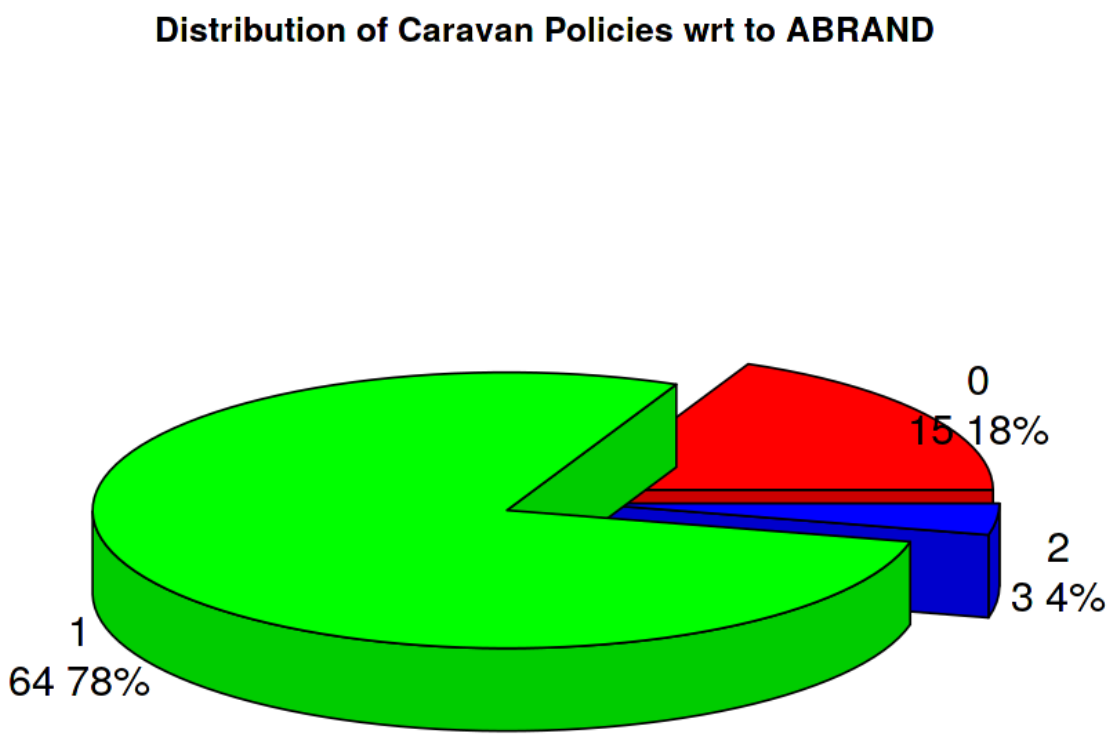
In [304]:

```
dist_ABRAND <- table(true_pos_data$X.ABRAND)
dist_ABRAND
```

```
 0  1  2
15 64  3
```

In [305]:

```
lbls <- paste(names(dist_ABRAND), "\n", dist_ABRAND, sep="")
pct <- round(dist_ABRAND/sum(dist_ABRAND)*100)
lbls <- paste(lbls, pct)
lbls <- paste(lbls,"%",sep="")
pie3D(dist_ABRAND,labels=lbls,explode=0.1,
   main="Distribution of Caravan Policies wrt to ABRAND")
```



We can observe from the pie chart the 78% of customers with 1 fire policy have bought the CARAVAN policy. Therefore, the marketing team should target customers with 1 fire policy.

Finally, we have explored all the features that have contributed towards the response feature. We have also identified the market segments the marketing should be targeting in order to reap maximum benefits and revenue

---

9. Conclusion

In Conclusion, we can say that, we have followed the most logical of methodologies to tackle this problem and have come up with plausible results. The due process was supplemented with Preliminary Analysis, Exploratory Data Analysis, In-depth Statistical Analsysis. The hypothesis was conformed with severl modeling techniques and statistical anlaysis. Additionally, the technique of choosing overlapping features has produced reliable results. Although this methodlogy seems to suit this dataset. This may not suit other datasets. Therefore,a big takeaway from this exercise is to employ a customisable methodology to tackle the problem and come up with reliable solution.

---

10. References

- *https://www.datacamp.com/community/tutorials/logistic-regression-R (https://www.datacamp.com/community/tutorials/logistic-regression-R)*
- *https://www.rdocumentation.org/packages/olsrr/versions/0.4.0/vignettes/variable_selection.Rmd (https://www.rdocumentation.org/packages/olsrr/versions/0.4.0/vignettes/variable_selection.Rmd)*
- *https://stats.stackexchange.com/questions/119835/correlation-between-a-nominal-iv-and-a-continuous-dv-variable/124618#124618 (https://stats.stackexchange.com/questions/119835/correlation-between-a-nominal-iv-and-a-continuous-dv-variable/124618#124618)*
- *https://www.quora.com/What-kind-of-test-should-I-use-to-test-the-correlation-between-categorical-and-numeric-variables (https://www.quora.com/What-kind-of-test-should-I-use-to-test-the-correlation-between-categorical-and-numeric-variables)*
- *https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc (https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc)*
- *https://machinelearningmastery.com/an-introduction-to-feature-selection/ (https://machinelearningmastery.com/an-introduction-to-feature-selection/)*
- *https://www.displayr.com/linear-discriminant-analysis-in-r-an-introduction/ (https://www.displayr.com/linear-discriminant-analysis-in-r-an-introduction/)*
- *https://rpubs.com/gabrielmartos/discriminantR (https://rpubs.com/gabrielmartos/discriminantR)*
- *https://www.datacamp.com/community/tutorials/feature-selection-R-boruta (https://www.datacamp.com/community/tutorials/feature-selection-R-boruta)*
- *https://rpubs.com/ifn1411/LDA (https://rpubs.com/ifn1411/LDA)*
- *https://towardsdatascience.com/linear-discriminant-analysis-lda-101-using-r-6a97217a55a6 (https://towardsdatascience.com/linear-discriminant-analysis-lda-101-using-r-6a97217a55a6)*
- *https://www.kdnuggets.com/2015/05/7-methods-data-dimensionality-reduction.html (https://www.kdnuggets.com/2015/05/7-methods-data-dimensionality-reduction.html)*
- *https://medium.com/@outside2SDs/an-overview-of-correlation-measures-between-categorical-and-continuous-variables-4c7f85610365 (https://medium.com/@outside2SDs/an-overview-of-correlation-measures-between-categorical-and-continuous-variables-4c7f85610365)*
- *https://medium.com/@outside2SDs/an-overview-of-correlation-measures-between-categorical-and-continuous-variables-4c7f85610365 (https://medium.com/@outside2SDs/an-overview-of-correlation-measures-between-categorical-and-continuous-variables-4c7f85610365)*