German Credit Report

We import the wine dataset from the xls file 'German Credit.xls' and name our dataframe "credit".

```
library(readxl)
credit <- read_xls("German Credit.xls")
credit <-  na.omit(credit)
```

Problem 5, Part a:

According to the dataset description, a lot of variables are infact categorical but are present as numerical variables as seen from the data summary. So we convert these variables into factor variable.

```
summary(credit)

##       OBS#           CHK_ACCT        DURATION        HISTORY
##  Min.   :   1.0   Min.   :0.000   Min.   : 4.0   Min.   :0.000
##  1st Qu.: 250.8   1st Qu.:0.000   1st Qu.:12.0   1st Qu.:2.000
##  Median : 500.5   Median :1.000   Median :18.0   Median :2.000
##  Mean   : 500.5   Mean   :1.577   Mean   :20.9   Mean   :2.545
##  3rd Qu.: 750.2   3rd Qu.:3.000   3rd Qu.:24.0   3rd Qu.:4.000
##  Max.   :1000.0   Max.   :3.000   Max.   :72.0   Max.   :4.000
##     NEW_CAR          USED_CAR        FURNITURE        RADIO/TV
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.00
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.00
##  Median :0.000   Median :0.000   Median :0.000   Median :0.00
##  Mean   :0.234   Mean   :0.103   Mean   :0.181   Mean   :0.28
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:1.00
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.00
##    EDUCATION      RETRAINING        AMOUNT          SAV_ACCT
##  Min.   :0.00   Min.   :0.000   Min.   :  250   Min.   :0.000
##  1st Qu.:0.00   1st Qu.:0.000   1st Qu.: 1366   1st Qu.:0.000
##  Median :0.00   Median :0.000   Median : 2320   Median :0.000
##  Mean   :0.05   Mean   :0.097   Mean   : 3271   Mean   :1.105
##  3rd Qu.:0.00   3rd Qu.:0.000   3rd Qu.: 3972   3rd Qu.:2.000
##  Max.   :1.00   Max.   :1.000   Max.   :18424   Max.   :4.000
##    EMPLOYMENT     INSTALL_RATE       MALE_DIV      MALE_SINGLE
##  Min.   :0.000   Min.   :1.000   Min.   :0.00   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:0.00   1st Qu.:0.000
##  Median :2.000   Median :3.000   Median :0.00   Median :1.000
##  Mean   :2.384   Mean   :2.973   Mean   :0.05   Mean   :0.548
##  3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:0.00   3rd Qu.:1.000
##  Max.   :4.000   Max.   :4.000   Max.   :1.00   Max.   :1.000
##  MALE_MAR_or_WID  CO-APPLICANT      GUARANTOR     PRESENT_RESIDENT
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :1.000
```

```
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:2.000
##  Median :0.000   Median :0.000   Median :0.000   Median :3.000
##  Mean   :0.092   Mean   :0.041   Mean   :0.052   Mean   :2.845
##  3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:0.000   3rd Qu.:4.000
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :4.000
##   REAL_ESTATE    PROP_UNKN_NONE       AGE        OTHER_INSTALL
##  Min.   :0.000   Min.   :0.000   Min.   :19.00   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:27.00   1st Qu.:0.000
##  Median :0.000   Median :0.000   Median :33.00   Median :0.000
##  Mean   :0.282   Mean   :0.154   Mean   :35.55   Mean   :0.186
##  3rd Qu.:1.000   3rd Qu.:0.000   3rd Qu.:42.00   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.000   Max.   :75.00   Max.   :1.000
##       RENT          OWN_RES       NUM_CREDITS         JOB
##  Min.   :0.000   Min.   :0.000   Min.   :1.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:2.000
##  Median :0.000   Median :1.000   Median :1.000   Median :2.000
##  Mean   :0.179   Mean   :0.713   Mean   :1.407   Mean   :1.904
##  3rd Qu.:0.000   3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :1.000   Max.   :1.000   Max.   :4.000   Max.   :3.000
##  NUM_DEPENDENTS    TELEPHONE        FOREIGN         RESPONSE
##  Min.   :1.000   Min.   :0.000   Min.   :0.000   Min.   :0.0
##  1st Qu.:1.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.0
##  Median :1.000   Median :0.000   Median :0.000   Median :1.0
##  Mean   :1.155   Mean   :0.404   Mean   :0.037   Mean   :0.7
##  3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:0.000   3rd Qu.:1.0
##  Max.   :2.000   Max.   :1.000   Max.   :1.000   Max.   :1.0
```

```r
credit <- credit[,-1]

credit$CHK_ACCT <- factor(credit$CHK_ACCT)
credit$HISTORY <- factor(credit$HISTORY)
credit$NEW_CAR <- factor(credit$NEW_CAR)
credit$USED_CAR <- factor(credit$USED_CAR)
credit$FURNITURE <- factor(credit$FURNITURE)

credit$`RADIO/TV` <- factor(credit$`RADIO/TV`)
names(credit)[7]<-"Radio_TV"

credit$EDUCATION <- factor(credit$EDUCATION)
credit$RETRAINING <- factor(credit$RETRAINING)
credit$SAV_ACCT <- factor(credit$SAV_ACCT)
credit$EMPLOYMENT <- factor(credit$EMPLOYMENT)
credit$MALE_DIV <- factor(credit$MALE_DIV)
credit$MALE_SINGLE <- factor(credit$MALE_SINGLE)
credit$MALE_MAR_or_WID <- factor(credit$MALE_MAR_or_WID)
credit$`CO-APPLICANT` <- factor(credit$`CO-APPLICANT`)
```

```
names(credit)[17]<-"Coapplicant"
credit$GUARANTOR <- factor(credit$GUARANTOR)
credit$REAL_ESTATE <- factor(credit$REAL_ESTATE)
credit$PROP_UNKN_NONE <- factor(credit$PROP_UNKN_NONE)
credit$OTHER_INSTALL <- factor(credit$OTHER_INSTALL)
credit$RENT <- factor(credit$RENT)
credit$OWN_RES <- factor(credit$OWN_RES)
credit$JOB <- factor(credit$JOB)
credit$TELEPHONE <- factor(credit$TELEPHONE)
credit$FOREIGN <- factor(credit$FOREIGN)


credit <- credit %>%
 mutate(RESPONSE = plyr::mapvalues(RESPONSE, c(1,0), c("Good", "Bad"))
)
credit$RESPONSE <- factor(credit$RESPONSE)


credit %>%
  group_by(RESPONSE) %>%
  summarise(count = n(), 'proportion(in %)' = n()/1000*100)

## # A tibble: 2 x 3
##   RESPONSE count `proportion(in %)`
##   <fct>    <int>              <dbl>
## 1 Bad        300                 30
## 2 Good       700                 70
```

The proportion of Good and Bad respose is: 700 Good Response (70%) and 300 Bad Response (30%).

```
summary(credit)

##  CHK_ACCT    DURATION    HISTORY NEW_CAR USED_CAR FURNITURE Radio_T
V
##  0:274    Min.   : 4.0   0: 40   0:766   0:897    0:819     0:720
##  1:269    1st Qu.:12.0   1: 49   1:234   1:103    1:181     1:280
##  2: 63    Median :18.0   2:530
##  3:394    Mean   :20.9   3: 88
##           3rd Qu.:24.0   4:293
##           Max.   :72.0
##  EDUCATION RETRAINING     AMOUNT      SAV_ACCT EMPLOYMENT  INSTALL_
RATE
##  0:950     0:903      Min.   :  250   0:603    0: 62      Min.   :1
.000
##  1: 50     1: 97      1st Qu.: 1366   1:103    1:172      1st Qu.:2
.000
```

```
##                        Median : 2320   2: 63    2:339      Median :3
.000
##                        Mean   : 3271   3: 48    3:174      Mean   :2
.973
##                        3rd Qu.: 3972   4:183    4:253      3rd Qu.:4
.000
##                        Max.   :18424                       Max.   :4
.000
##  MALE_DIV MALE_SINGLE MALE_MAR_or_WID Coapplicant GUARANTOR
##  0:950    0:452       0:908           0:959       0:948
##  1: 50    1:548       1: 92           1: 41       1: 52
##
##
##
##
##  PRESENT_RESIDENT REAL_ESTATE PROP_UNKN_NONE      AGE         OTHER_
INSTALL
##  Min.   :1.000    0:718       0:846          Min.   :19.00   0:814
##  1st Qu.:2.000    1:282       1:154          1st Qu.:27.00   1:186
##  Median :3.000                               Median :33.00
##  Mean   :2.845                               Mean   :35.55
##  3rd Qu.:4.000                               3rd Qu.:42.00
##  Max.   :4.000                               Max.   :75.00
##  RENT    OWN_RES  NUM_CREDITS    JOB     NUM_DEPENDENTS  TELEPHONE
FOREIGN
##  0:821   0:287   Min.   :1.000   0: 22   Min.   :1.000   0:596
0:963
##  1:179   1:713   1st Qu.:1.000   1:200   1st Qu.:1.000   1:404
1: 37
##                  Median :1.000   2:630   Median :1.000
##                  Mean   :1.407   3:148   Mean   :1.155
##                  3rd Qu.:2.000           3rd Qu.:1.000
##                  Max.   :4.000           Max.   :2.000
##  RESPONSE
##  Bad :300
##  Good:700
##
##
##
##
```

Exploratory Data Analysis for Numerical Variables:

We have the following numerical variables:

DURATION

AMOUNT

INSTALL_RATE

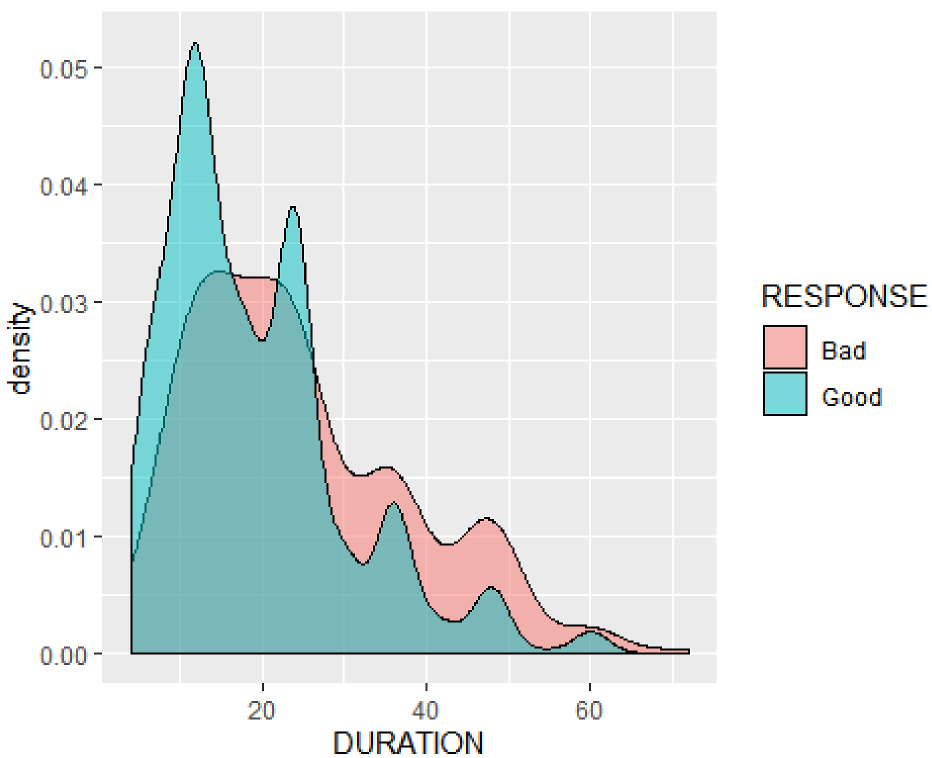
AGE
NUM_CREDITS
NUM_DEPENDENTS

I. DURATION:

```
credit %>%
group_by(RESPONSE) %>%
summarise(mean =mean(DURATION))
```

```
## # A tibble: 2 x 2
##   RESPONSE  mean
##   <fct>    <dbl>
## 1 Bad       24.9
## 2 Good      19.2
```

```
ggplot(data = credit, aes(DURATION, fill = RESPONSE)) +
  geom_density(alpha = 0.5)
```



Hypothesis Test to find if the difference in mean duration for Good credit and bad credit is statistically significant. (Independent Sampled T-test)

Null Hypothesis: mean duration (Good response) = mean duration(Bad Response)
Alternative: mean duration (Good response) != mean duration(Bad Response)

```r
t.test(DURATION ~ RESPONSE, data = credit, var.equal=FALSE, paired=FAL
SE)
```

```
##
##  Welch Two Sample t-test
##
## data:  DURATION by RESPONSE
## t = 6.4696, df = 485.44, p-value = 2.404e-10
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  3.936033 7.369682
## sample estimates:
##  mean in group Bad mean in group Good
##           24.86000           19.20714
```

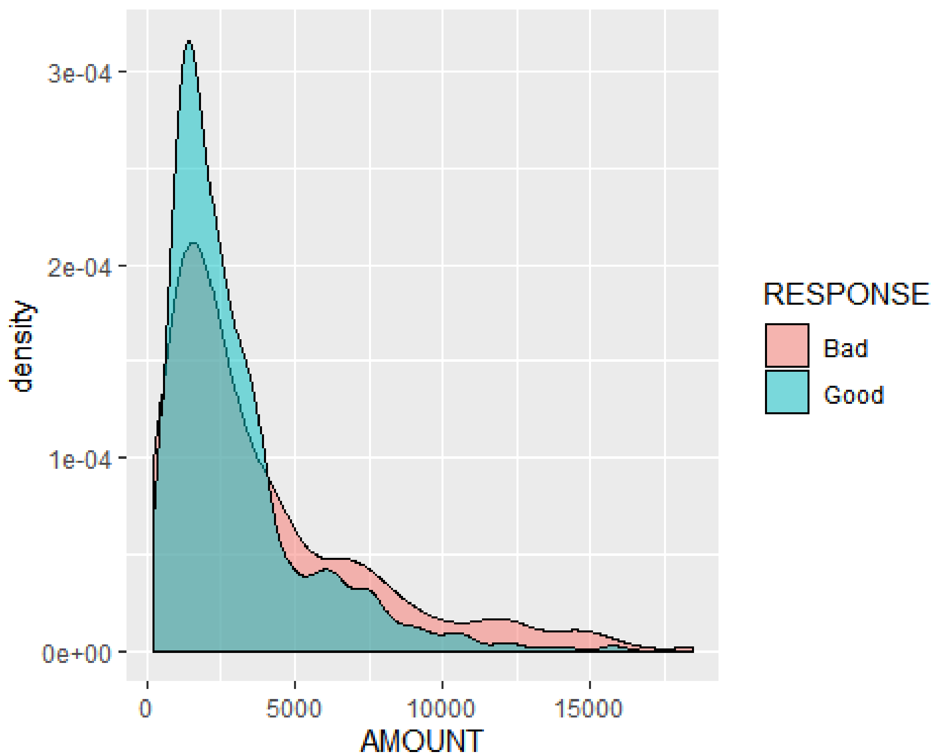
Finding: *The p-value for Welch two sample T-test is quite small. Hence we can conclude that the sample provides sufficient evidence that credit Response is dependent on duration of credit.*

II. AMOUNT:

```r
 credit %>%
  group_by(RESPONSE) %>%
  summarise(mean =mean(AMOUNT))
```

```
## # A tibble: 2 x 2
##   RESPONSE  mean
##   <fct>    <dbl>
## 1 Bad      3938.
## 2 Good     2985.
```

```r
ggplot(data = credit, aes(AMOUNT, fill = RESPONSE)) +
  geom_density(alpha = 0.5)
```



Hypothesis Test to find if the difference in mean amount for Good credit and bad credit is statistically significant. (Independent Sampled T-test)

Null Hypothesis: mean amount (Good response) = mean amount(Bad Response)
Alternative: mean amount (Good response) != mean amount(Bad Response)

```
t.test(AMOUNT ~ RESPONSE, data = credit, var.equal=FALSE, paired=FALSE
)

##
##  Welch Two Sample t-test
##
## data:  AMOUNT by RESPONSE
## t = 4.2642, df = 421.86, p-value = 2.478e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   513.534 1391.805
## sample estimates:
##  mean in group Bad mean in group Good
##           3938.127           2985.457
```

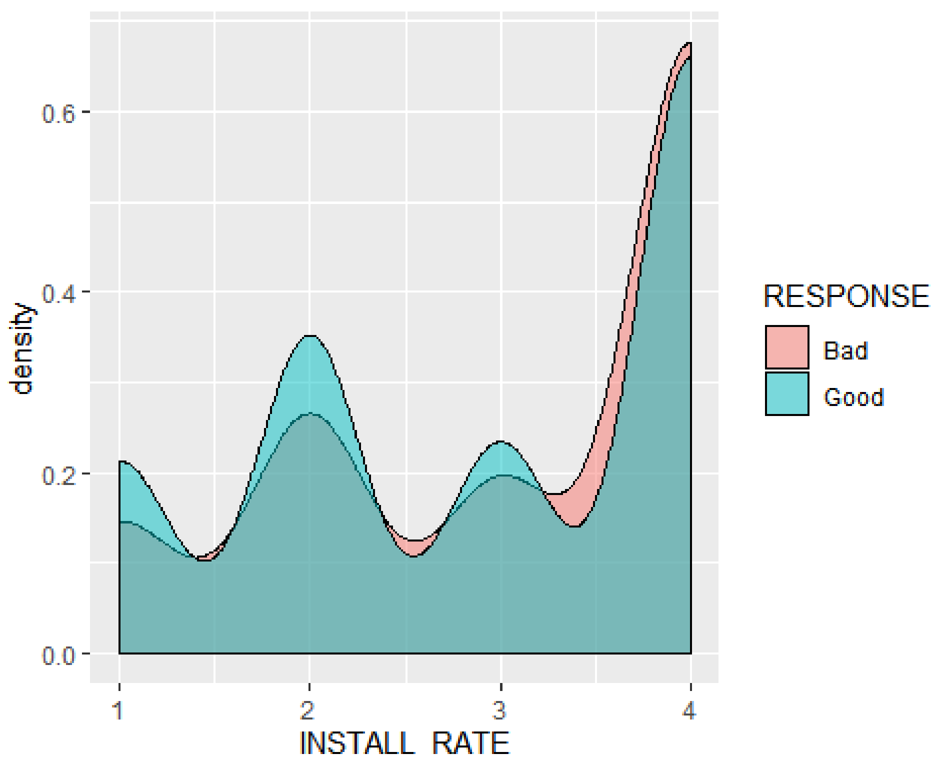
Finding: *The p-value for Welch two sample T-test is quite small. Hence we can conclude that the sample provides sufficient evidence that credit Response is dependent on credit amount.*

III. INSTALL_RATE:

Hypothesis Test to find if the difference in mean install_rate for Good credit and bad credit is statistically significant. (Independent Sampled T-test)

Null Hypothesis: mean install_rate (Good response) = mean install_rate(Bad Response)
Alternative: mean install_rate (Good response) != mean install_rate(Bad Response)

```
credit %>%
  group_by(RESPONSE) %>%
  summarise(mean =mean(INSTALL_RATE))

## # A tibble: 2 x 2
##   RESPONSE  mean
##   <fct>    <dbl>
## 1 Bad       3.10
## 2 Good      2.92

ggplot(data = credit, aes(INSTALL_RATE, fill = RESPONSE)) +
  geom_density(alpha = 0.5)
```




```
t.test(INSTALL_RATE ~ RESPONSE, data = credit, var.equal=FALSE, paired
=FALSE)

##
##  Welch Two Sample t-test
##
## data:  INSTALL_RATE by RESPONSE
```

```
## t = 2.3265, df = 584.68, p-value = 0.02034
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.0275216 0.3258117
## sample estimates:
##  mean in group Bad mean in group Good
##           3.096667           2.920000
```

Finding: *The actuall sample install_rate mean for Good and Bad response are quite close to one another. But the p-value for Welch two sample T-test comes out to be smaller than 0.05. Hence we can conclude that the sample provides sufficient evidence that credit Response is dependent on installment rate.*

IV. AGE:

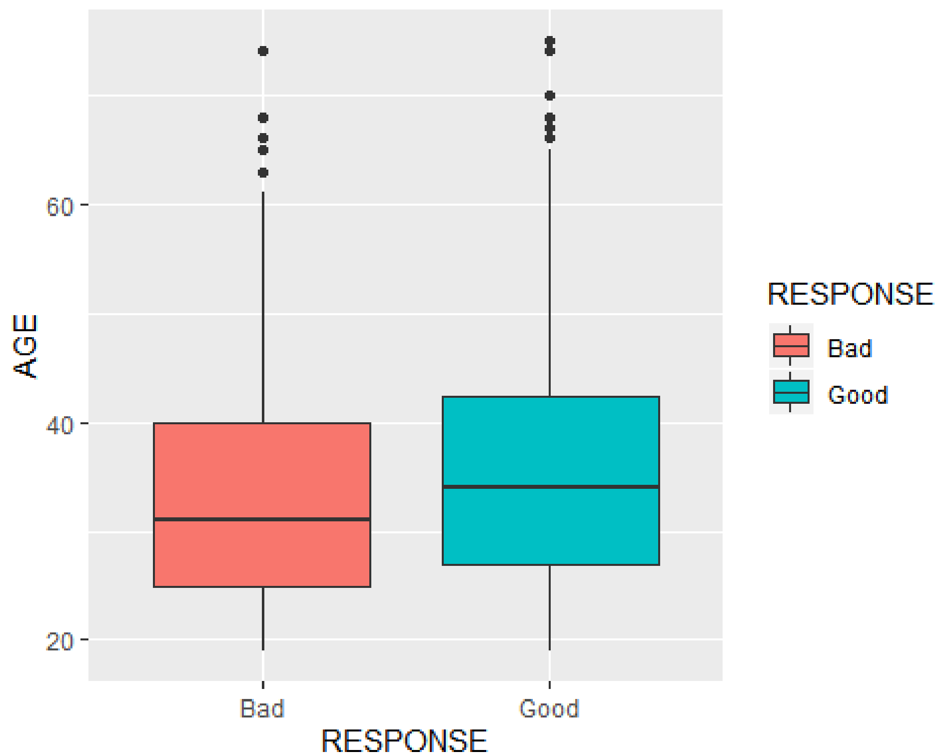
Hypothesis Test to find if the difference in mean age for Good credit and bad credit is statistically significant. (Independent Sampled T-test)

Null Hypothesis: mean age (Good response) = mean age(Bad Response)
Alternative: mean age(Good response) != mean age(Bad Response)

```
credit %>%
  group_by(RESPONSE) %>%
  summarise(mean =mean(AGE))
```

```
## # A tibble: 2 x 2
##   RESPONSE  mean
##   <fct>    <dbl>
## 1 Bad       34.0
## 2 Good      36.2
```

```
ggplot(data = credit, aes(x = RESPONSE, y = AGE, fill = RESPONSE)) +
  geom_boxplot()
```




```
t.test(AGE ~ RESPONSE, data = credit, var.equal=FALSE, paired=FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  AGE by RESPONSE
## t = -2.9072, df = 573.06, p-value = 0.003788
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -3.7884832 -0.7334216
## sample estimates:
##  mean in group Bad mean in group Good
##           33.96333           36.22429
```

Finding: *The actuall sample age mean for Good and Bad Response are quite close to one another. But the p-value for Welch two sample T-test comes out to be smaller than 0.05. Hence we can conclude that the sample provides sufficient evidence that credit Response is dependent on age.*

V. NUM_CREDITS:

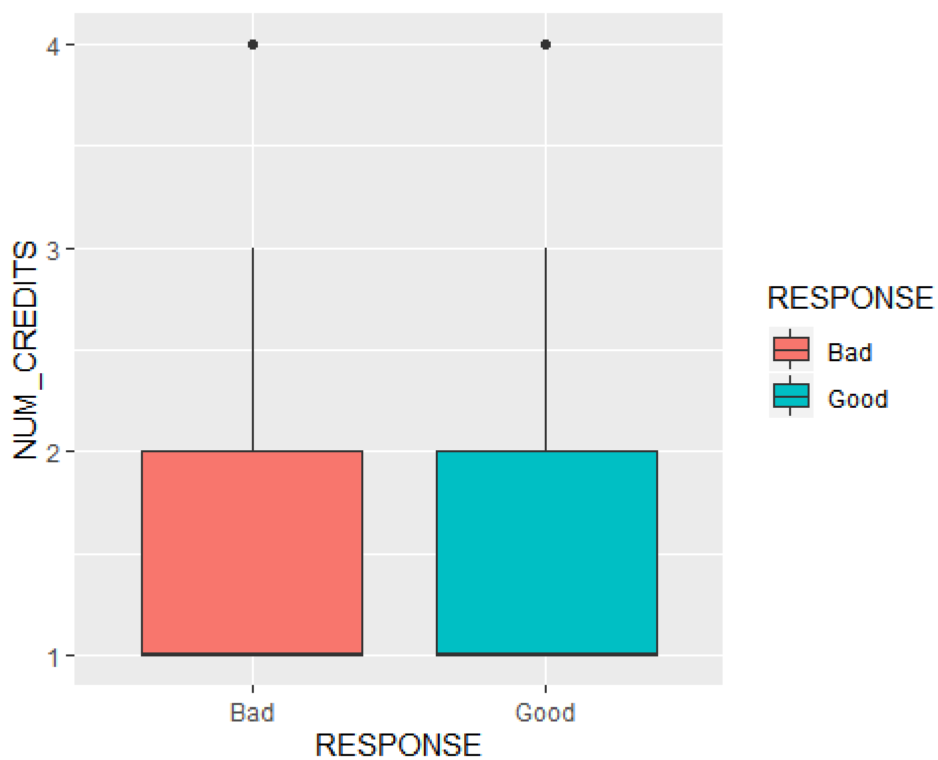
Hypothesis Test to find if the difference in mean number of credits for Good credit and bad credit is statistically significant. (Independent Sampled T-test)

Null Hypothesis: mean num_credits (Good response) = mean num_credits(Bad Response)
Alternative: mean num_credits(Good response) != mean num_credits(Bad Response)

```r
credit %>%
  group_by(RESPONSE) %>%
  summarise(mean =mean(NUM_CREDITS))
```

```
## # A tibble: 2 x 2
##   RESPONSE  mean
##   <fct>    <dbl>
## 1 Bad       1.37
## 2 Good      1.42
```

```r
ggplot(data = credit, aes(x = RESPONSE, y = NUM_CREDITS, fill = RESPON
SE)) +
  geom_boxplot()
```




```r
t.test(NUM_CREDITS ~ RESPONSE, data = credit, var.equal=FALSE, paired=
FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  NUM_CREDITS by RESPONSE
## t = -1.4718, df = 589, p-value = 0.1416
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.13450777  0.01926968
```

```
## sample estimates:
##  mean in group Bad mean in group Good
##          1.366667          1.424286
```

Finding: *The actuall sample num_credits mean for Good and Bad Response are quite close to one another. And the p-value for Welch two sample T-test comes out to be greater than 0.05. Hence we fail to reject the Null hypothesis and we can say that number of credits does not have a statistically significant association with credit Response.*

VI. NUM_DEPENDENTS:

```
credit %>%
  group_by(RESPONSE) %>%
  summarise(mean =mean(NUM_DEPENDENTS))
```

```
## # A tibble: 2 x 2
##   RESPONSE  mean
##   <fct>    <dbl>
## 1 Bad       1.15
## 2 Good      1.16
```

```
t.test(NUM_DEPENDENTS ~ RESPONSE, data = credit, var.equal=FALSE, pair
ed=FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  NUM_DEPENDENTS by RESPONSE
## t = -0.095447, df = 568.51, p-value = 0.924
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.05137712  0.04661522
## sample estimates:
##  mean in group Bad mean in group Good
##          1.153333          1.155714
```

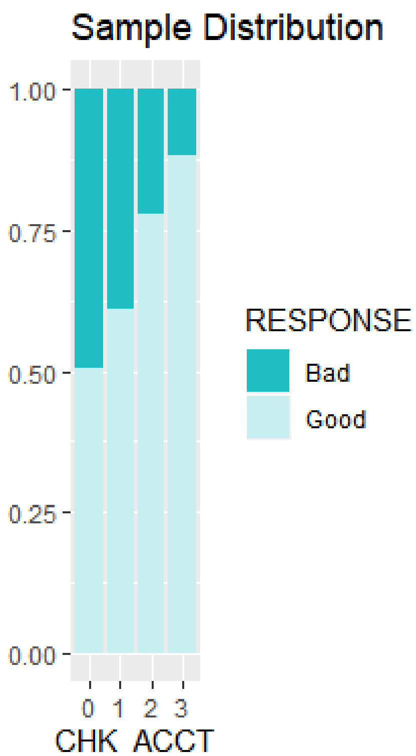
Finding: *The p-value for Welch two sample T-test comes out to be greater than 0.05. Hence we fail to reject the Null hypothesis and we can say that number of dependents does not have a statistically significant association with credit Response.*

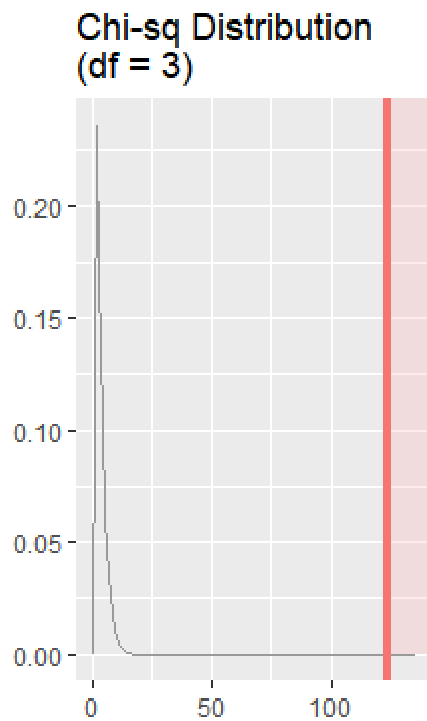
Exploratory Data Analysis for Categorical Variables:
We will study the association of Response with a categorical variables using chi-square tests:

I. CHK_ACCT

```
inference(data = credit,
          x= CHK_ACCT,
          y=RESPONSE,
```

```
          statistic = "proportion",
          type = "ht",
          null = 0,
          alternative = "greater",
          method = "theoretical")

## Warning: Ignoring null value since it's undefined for chi-square te
st of
## independence

## Response variable: categorical (2 levels)
## Explanatory variable: categorical (4 levels)
## Observed:
##    y
## x   Bad Good
##   0 135  139
##   1 105  164
##   2  14   49
##   3  46  348
##
## Expected:
##    y
## x     Bad  Good
##   0  82.2 191.8
##   1  80.7 188.3
##   2  18.9  44.1
##   3 118.2 275.8
##
## H0: CHK_ACCT and RESPONSE are independent
## HA: CHK_ACCT and RESPONSE are dependent
## chi_sq = 123.7209, df = 3, p_value = 0
```



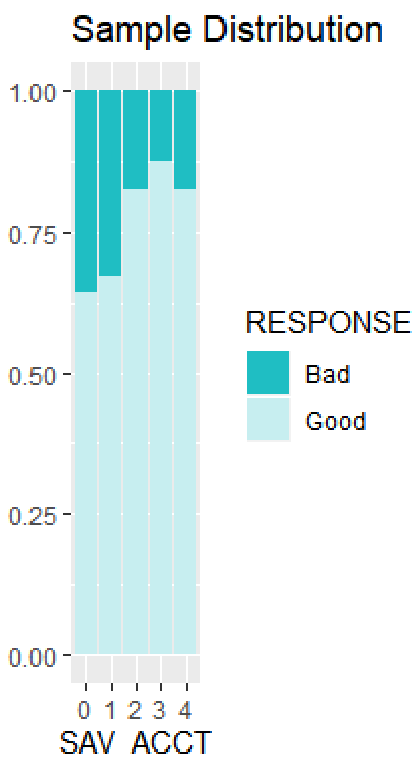


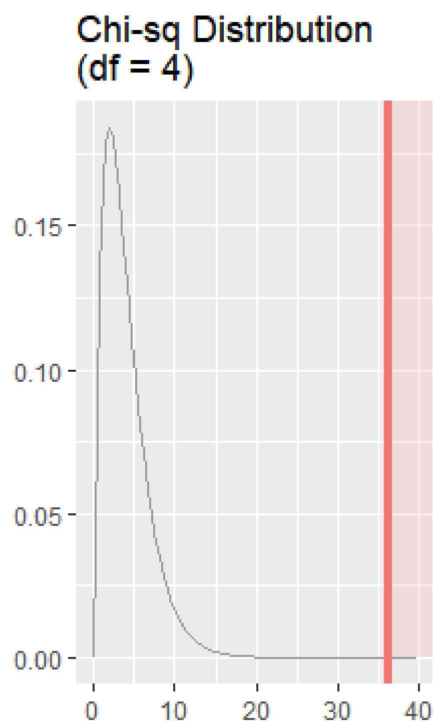
Finding: *The results of Chi Square goodness of fit test give a p-value of 0, which is less than 0.05. Hence, we will reject the null hypothesis. So, we can conclude that there is statistically significant evidence showing that CHK_ACCT and RESPONSE are dependent.*

II. SAV_ACCT

```
inference(data = credit,
          x= SAV_ACCT,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          alternative = "greater",
          method = "theoretical")

## Warning: Ignoring null value since it's undefined for chi-square te
st of
## independence

## Response variable: categorical (2 levels)
## Explanatory variable: categorical (5 levels)
## Observed:
##    y
## x    Bad Good
##   0 217  386
```

```
##   1  34   69
##   2  11   52
##   3   6   42
##   4  32  151
##
## Expected:
##    y
## x     Bad  Good
##   0 180.9 422.1
##   1  30.9  72.1
##   2  18.9  44.1
##   3  14.4  33.6
##   4  54.9 128.1
##
## H0: SAV_ACCT and RESPONSE are independent
## HA: SAV_ACCT and RESPONSE are dependent
## chi_sq = 36.0989, df = 4, p_value = 0
```



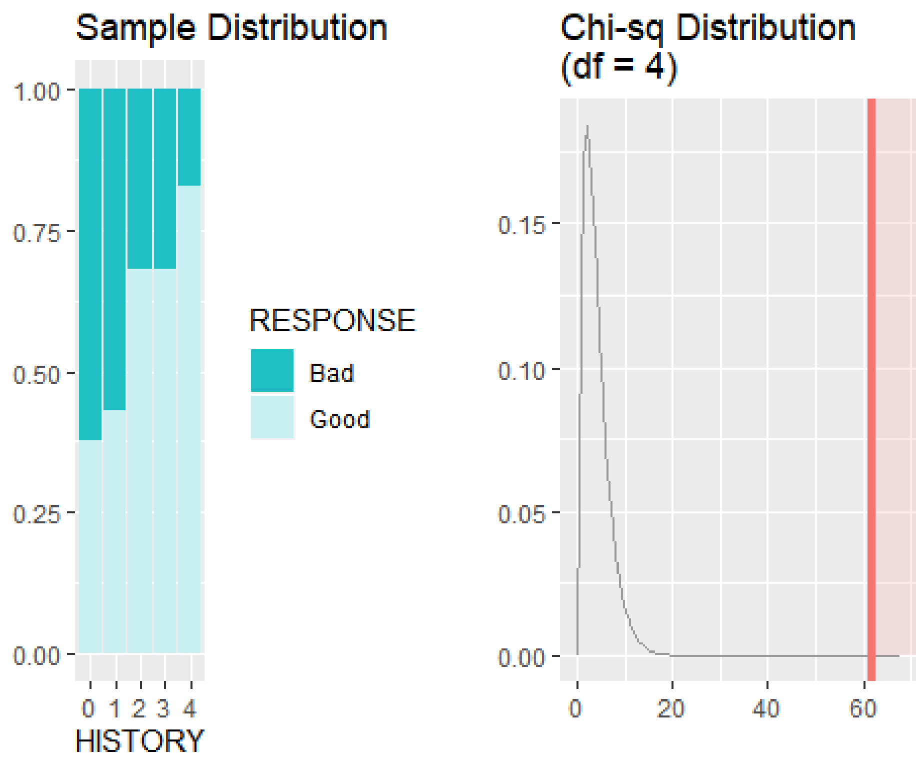


Finding: *The results of Chi Square goodness of fit test give a p-value of 0, which is less than 0.05. Hence, we will reject the null hypothesis. So, we can conclude that there is statistically significant evidence showing that SAV_ACCT and RESPONSE are dependent.*

III. HISTORY

```r
inference(data = credit,
          x= HISTORY,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          alternative = "greater",
          method = "theoretical")
```

```
## Warning: Ignoring null value since it's undefined for chi-square te
st of
## independence

## Response variable: categorical (2 levels)
## Explanatory variable: categorical (5 levels)
## Observed:
##     y
## x    Bad Good
##   0   25   15
##   1   28   21
##   2  169  361
##   3   28   60
##   4   50  243
##
## Expected:
##     y
## x      Bad   Good
##   0   12.0   28.0
##   1   14.7   34.3
##   2  159.0  371.0
##   3   26.4   61.6
##   4   87.9  205.1
##
## H0: HISTORY and RESPONSE are independent
## HA: HISTORY and RESPONSE are dependent
## chi_sq = 61.6914, df = 4, p_value = 0
```

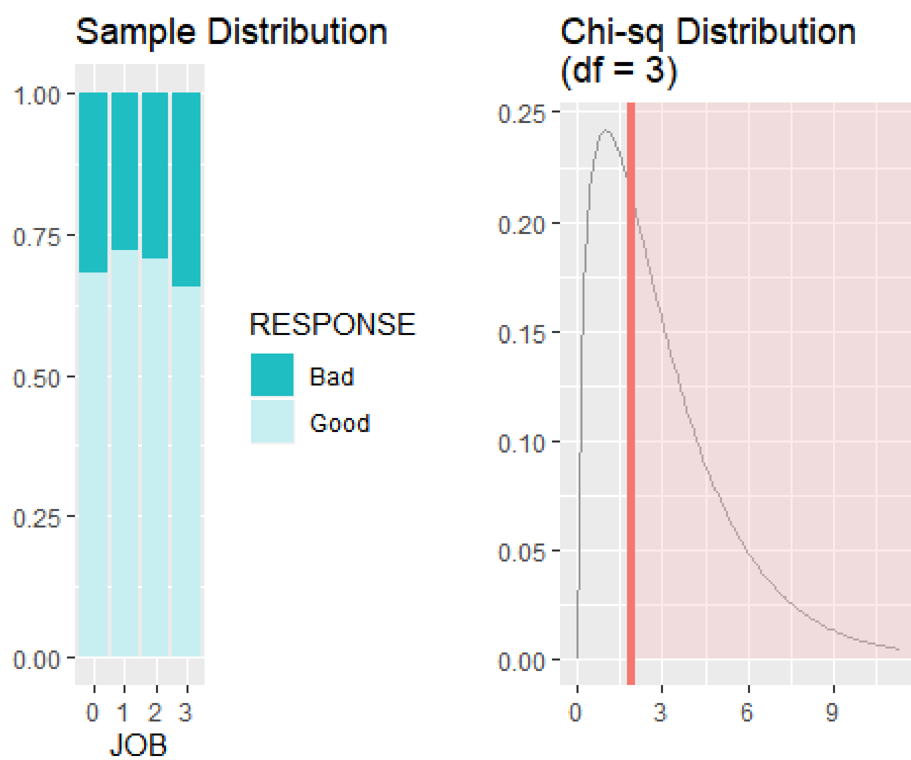


Finding: *The results of Chi Square goodness of fit test give a p-value of 0, which is less than 0.05. Hence, we will reject the null hypothesis. So, we can conclude that there is statistically significant evidence showing that HISTORY and RESPONSE are dependent.*

IV. JOB

```
inference(data = credit,
          x= JOB,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          alternative = "greater",
          method = "theoretical")

## Warning: Ignoring null value since it's undefined for chi-square te
st of
## independence

## Response variable: categorical (2 levels)
## Explanatory variable: categorical (4 levels)
## Observed:
##    y
## x    Bad Good
##   0   7   15
```

```
##    1   56   144
##    2  186   444
##    3   51    97
##
## Expected:
##     y
## x      Bad   Good
##    0    6.6   15.4
##    1   60.0  140.0
##    2  189.0  441.0
##    3   44.4  103.6
##
## H0: JOB and RESPONSE are independent
## HA: JOB and RESPONSE are dependent
## chi_sq = 1.8852, df = 3, p_value = 0.5966
```



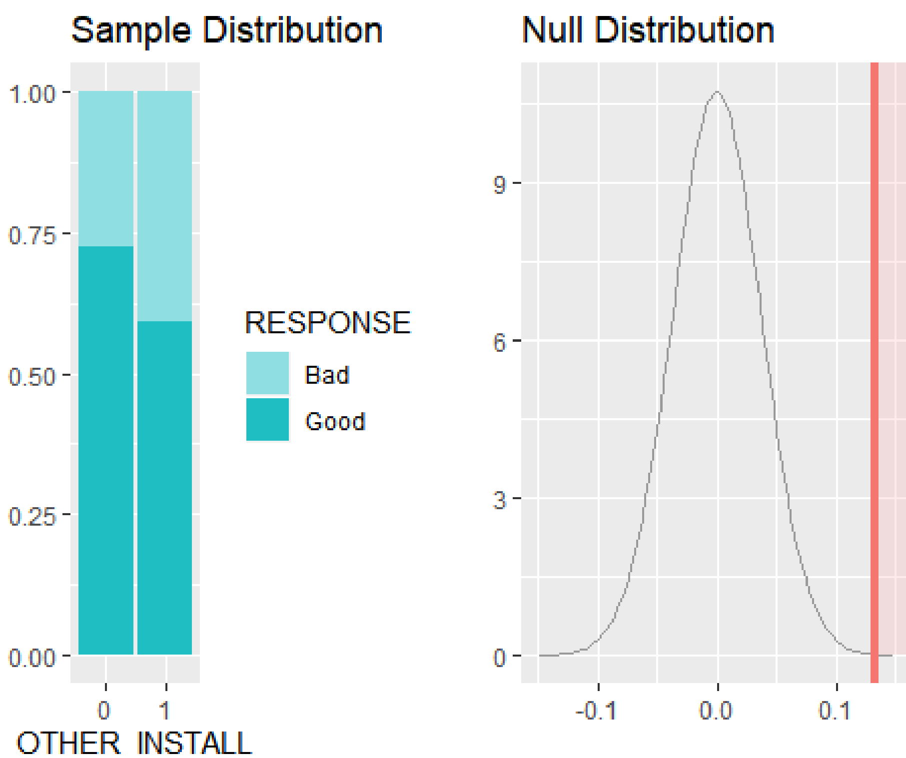
Finding: *The results of Chi Square goodness of fit test give a p-value of 0.59, which is greater than 0.05. Hence, we fail to reject the null hypothesis. So, we can conclude that there is not enough evidence showing association between JOB and RESPONSE.*

V. OTHER_INSTALL

```
inference(data = credit,
          x= OTHER_INSTALL,
          y=RESPONSE,
```

```
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "greater",
          method = "theoretical")
```

```
## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 814, p_hat_0 = 0.7248
## n_1 = 186, p_hat_1 = 0.5914
## H0: p_0 =  p_1
## HA: p_0 > p_1
## z = 3.5824
## p_value = 2e-04
```



Finding: *Null hypothesis suggests that proportion of people with Good response is equal for people with no other installments and those with othet installments. The results of hypothesis test give a p-value of less than 0.05. Hence, we can reject the null hypothesis, and conclude that proportion of people with Good Response is higher for people with no Other installments.*

VI. FOREIGN

```
inference(data = credit,
          x= FOREIGN,
```

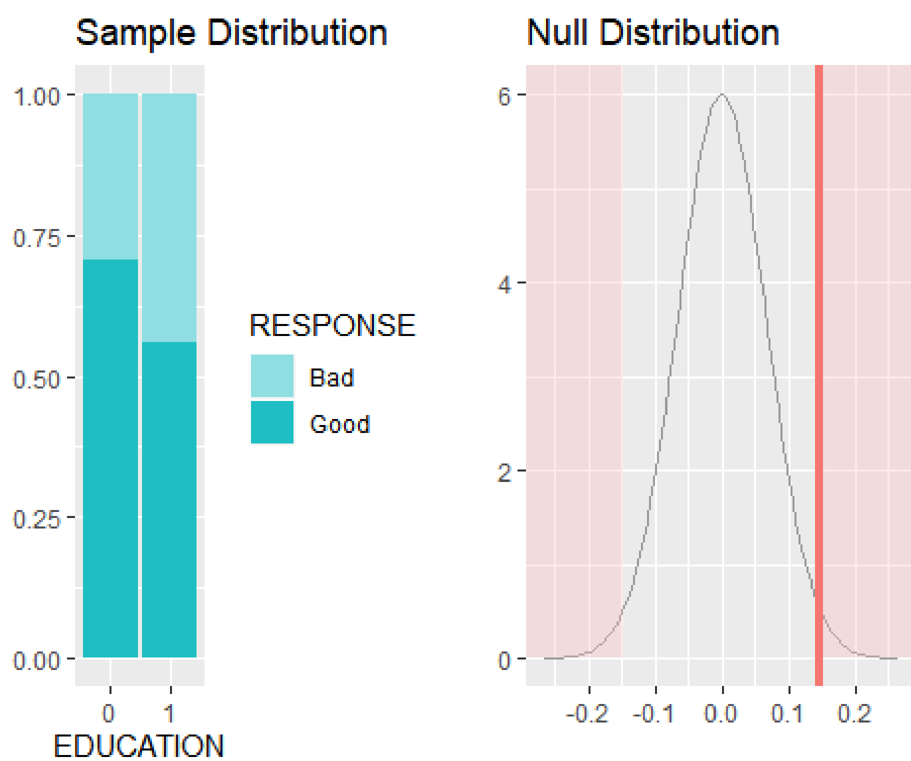
```
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 963, p_hat_0 = 0.6926
## n_1 = 37, p_hat_1 = 0.8919
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -2.5956
## p_value = 0.0094
```

Finding: *Null hypothesis suggests that proportion of people with Good response is equal for Foreign workers and non-foreign workers. The results of hypothesis test give a p-value of less than 0.05. Hence, we can reject the null hypothesis, and conclude that proportion of people with Good Response is not euqal for Foreign workers and non-foreign workers.*

Similarly we do for all other variables:

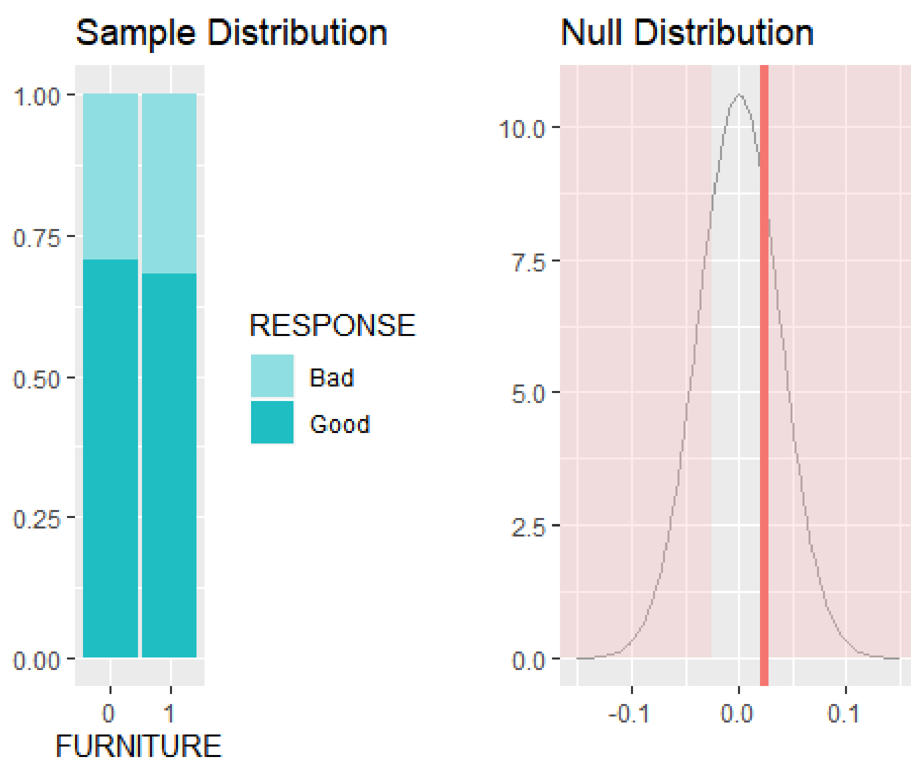
```r
inference(data = credit,
          x= EDUCATION,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")
```

```
## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 950, p_hat_0 = 0.7074
## n_1 = 50, p_hat_1 = 0.56
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 2.2164
## p_value = 0.0267
```



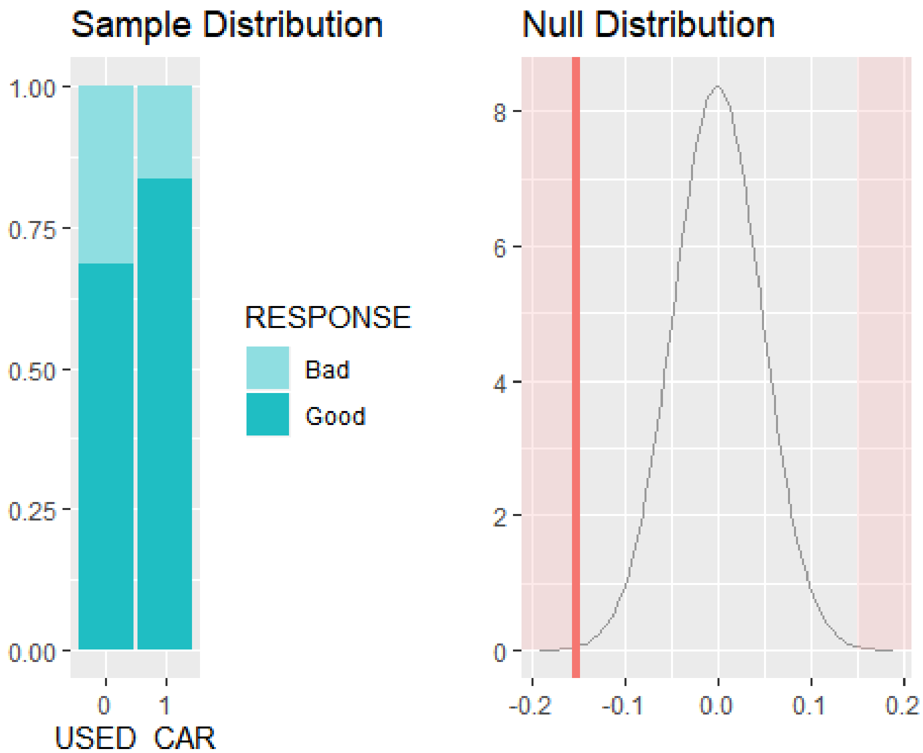

```r
inference(data = credit,
          x= FURNITURE,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
```

```
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 819, p_hat_0 = 0.7045
## n_1 = 181, p_hat_1 = 0.6796
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 0.6631
## p_value = 0.5072
```



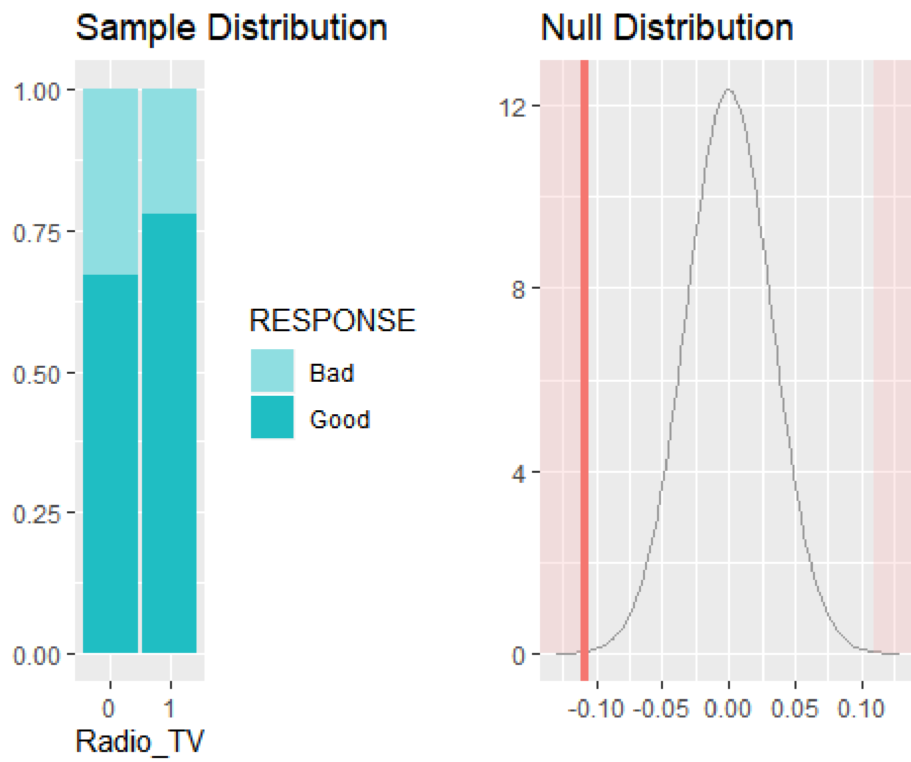

```
inference(data = credit,
          x= USED_CAR,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
```

```
## n_0 = 897, p_hat_0 = 0.6845
## n_1 = 103, p_hat_1 = 0.835
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -3.1557
## p_value = 0.0016
```




```
inference(data = credit,
          x= Radio_TV,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")
```

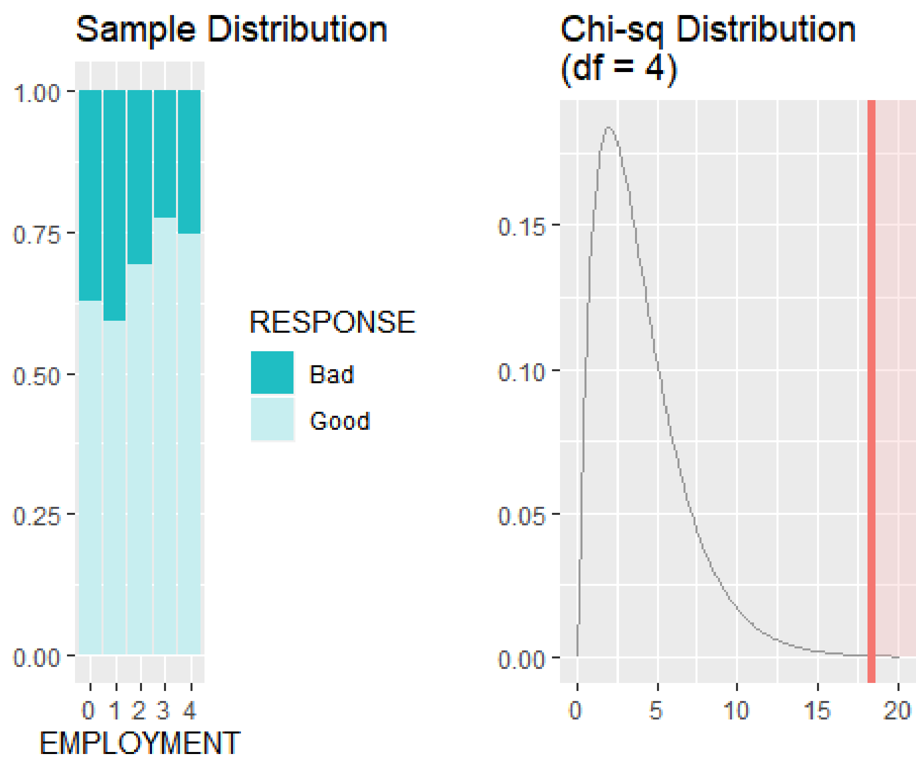
```
## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 720, p_hat_0 = 0.6694
## n_1 = 280, p_hat_1 = 0.7786
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -3.3812
## p_value = 7e-04
```




```r
inference(data = credit,
          x= EMPLOYMENT,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "greater",
          method = "theoretical")
```

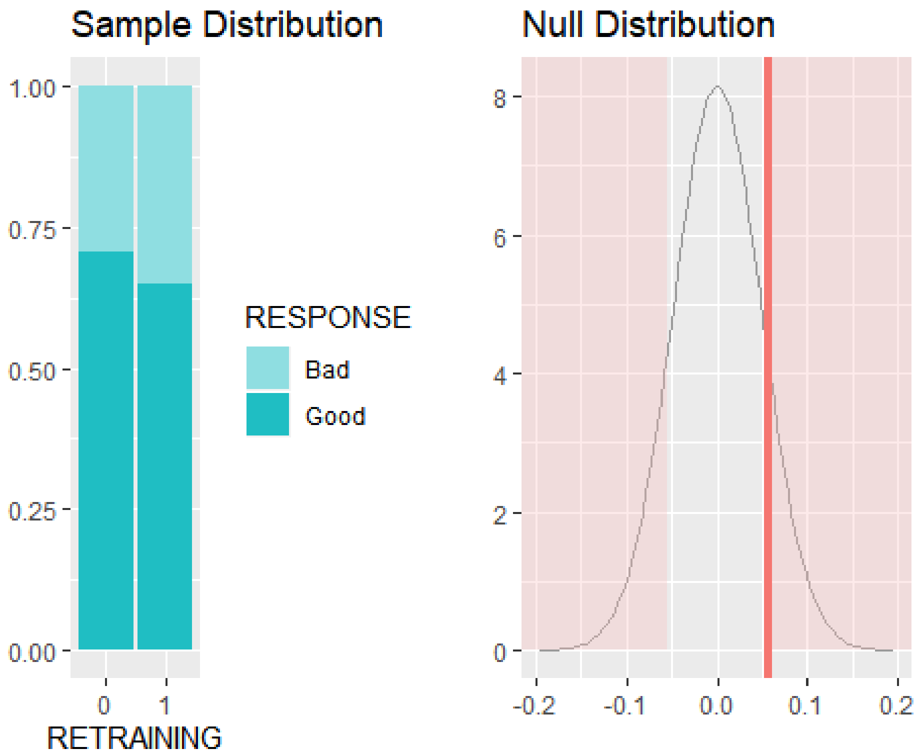
```
## Warning: Ignoring null value since it's undefined for chi-square te
st of
## independence

## Response variable: categorical (2 levels)
## Explanatory variable: categorical (5 levels)
## Observed:
##    y
## x   Bad Good
##   0  23   39
##   1  70  102
##   2 104  235
##   3  39  135
##   4  64  189
##
```

```
## Expected:
##    y
## x      Bad  Good
##   0  18.6  43.4
##   1  51.6 120.4
##   2 101.7 237.3
##   3  52.2 121.8
##   4  75.9 177.1
##
## H0: EMPLOYMENT and RESPONSE are independent
## HA: EMPLOYMENT and RESPONSE are dependent
## chi_sq = 18.3683, df = 4, p_value = 0.001
```

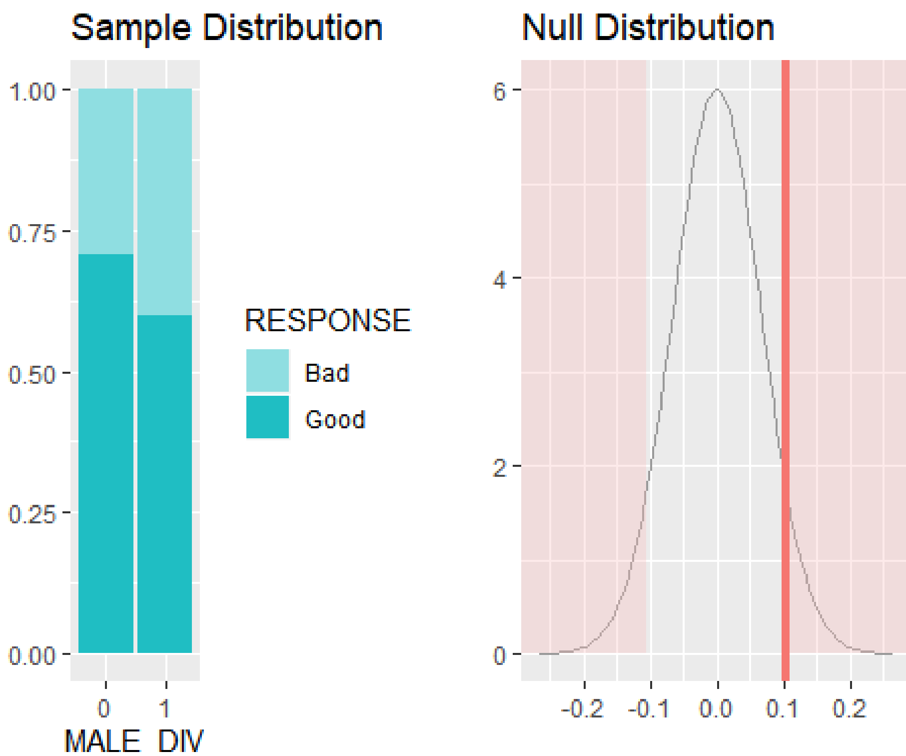



```r
inference(data = credit,
          x= RETRAINING,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
```

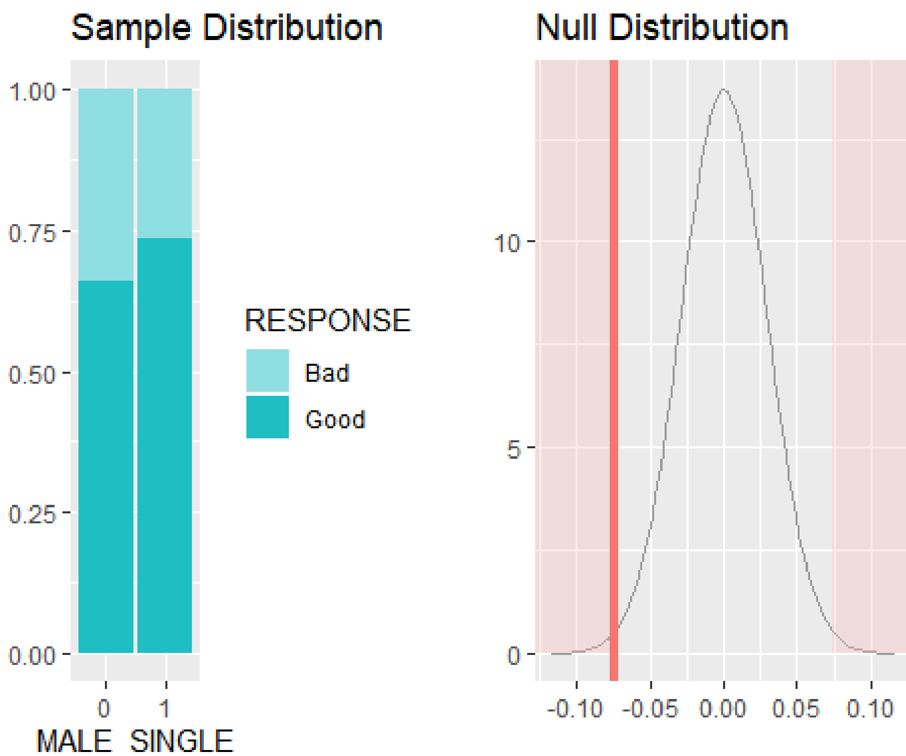
```
## n_0 = 903, p_hat_0 = 0.7054
## n_1 = 97, p_hat_1 = 0.6495
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 1.1425
## p_value = 0.2532
```



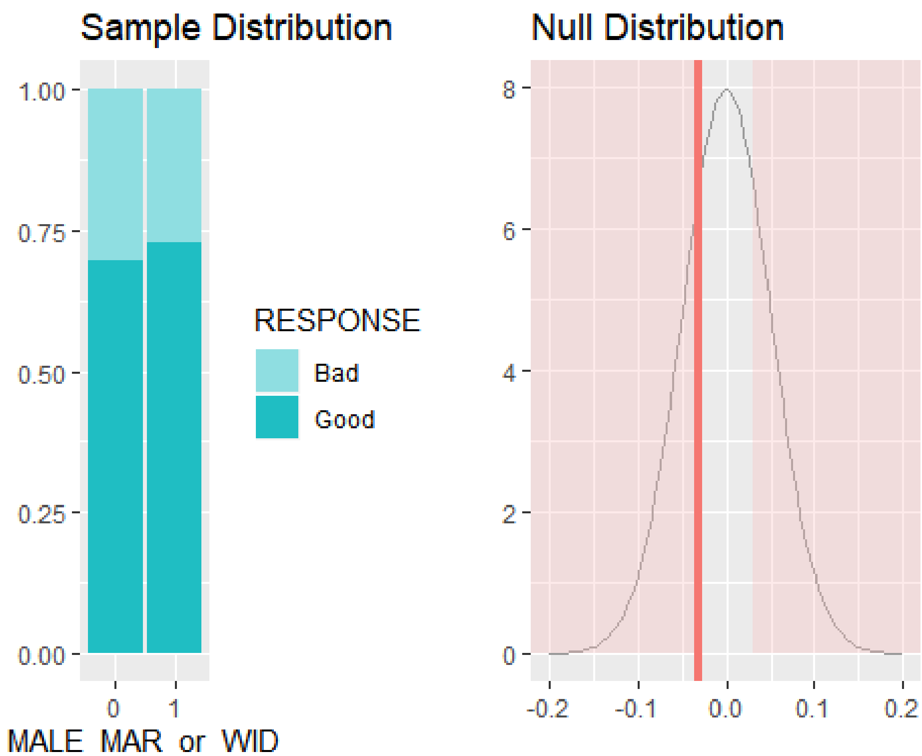

```
inference(data = credit,
          x= MALE_DIV,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 950, p_hat_0 = 0.7053
## n_1 = 50, p_hat_1 = 0.6
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 1.5831
## p_value = 0.1134
```




```r
inference(data = credit,
          x= MALE_SINGLE,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 452, p_hat_0 = 0.6593
## n_1 = 548, p_hat_1 = 0.7336
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -2.5512
## p_value = 0.0107
```

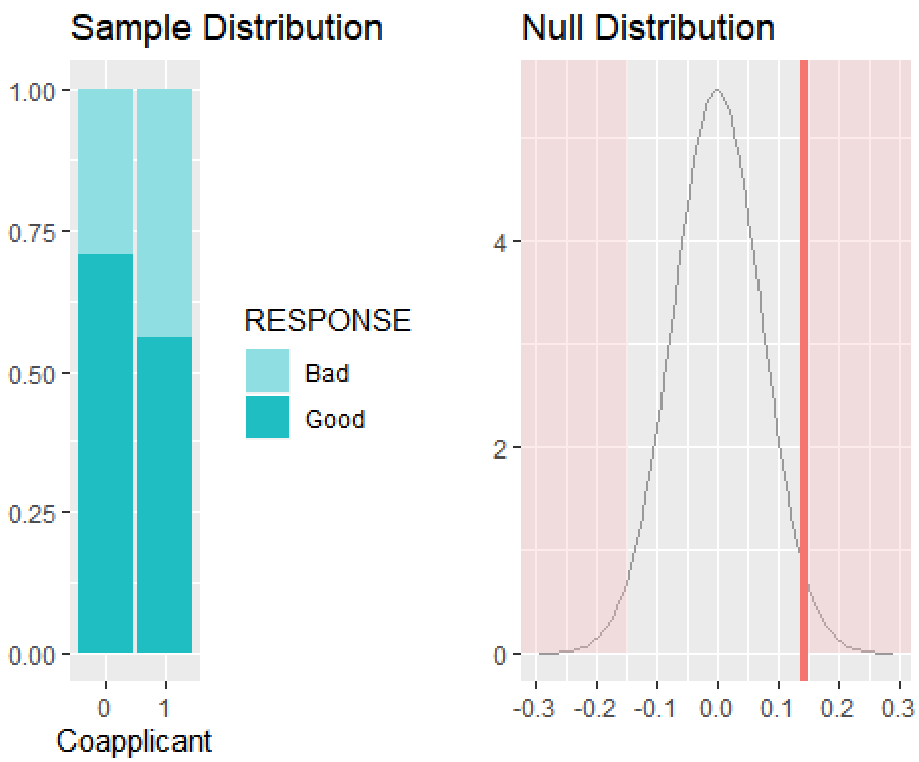



```
inference(data = credit,
          x= MALE_MAR_or_WID,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 908, p_hat_0 = 0.6971
## n_1 = 92, p_hat_1 = 0.7283
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -0.6208
## p_value = 0.5348
```



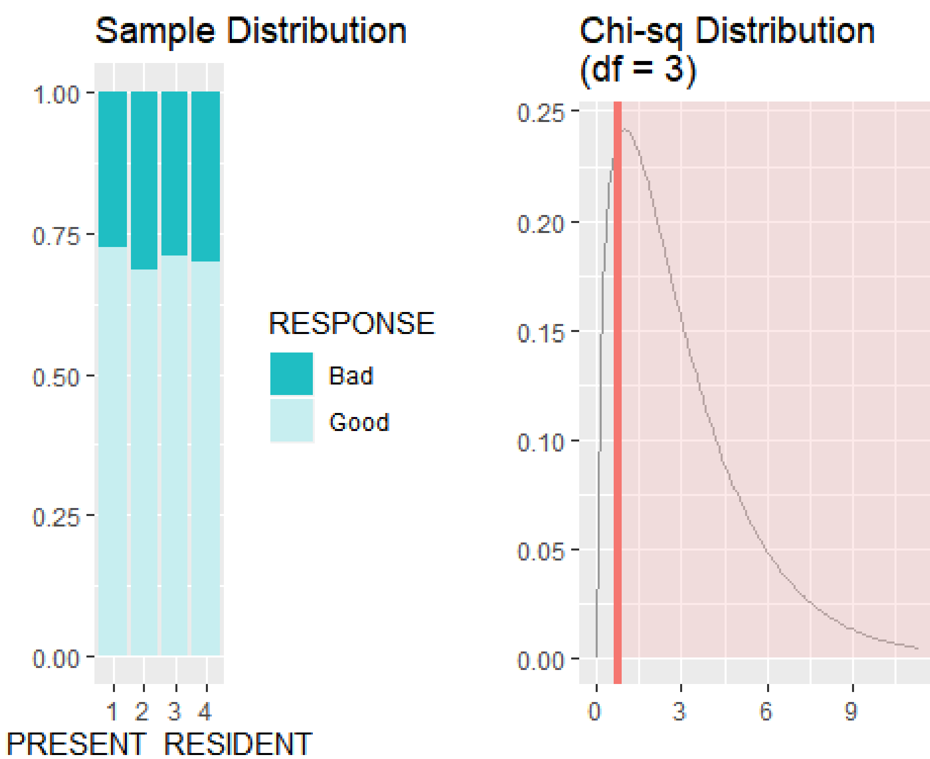

```
inference(data = credit,
          x= Coapplicant,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")
```

```
## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 959, p_hat_0 = 0.7059
## n_1 = 41, p_hat_1 = 0.561
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 1.9836
## p_value = 0.0473
```



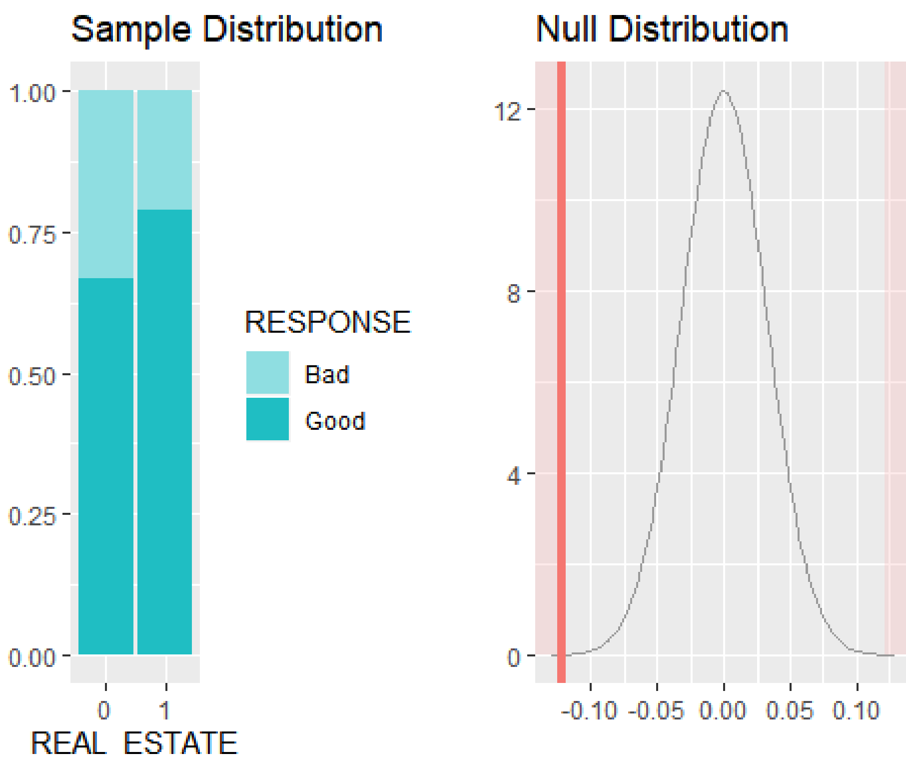

```
inference(data = credit,
          x= PRESENT_RESIDENT,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "greater",
          method = "theoretical")

## Warning: Explanatory variable was numerical, it has been converted
##                  to categorical. In order to avoid this warning, first
convert
##                  your explanatory variable to a categorical variable u
sing the
##                  as.factor() function

## Warning: Ignoring null value since it's undefined for chi-square te
st of independence

## Response variable: categorical (2 levels)
## Explanatory variable: categorical (4 levels)
## Observed:
##    y
## x   Bad Good
```

```
##    1  36   94
##    2  97  211
##    3  43  106
##    4 124  289
##
## Expected:
##     y
## x      Bad  Good
##    1  39.0  91.0
##    2  92.4 215.6
##    3  44.7 104.3
##    4 123.9 289.1
##
## H0: PRESENT_RESIDENT and RESPONSE are independent
## HA: PRESENT_RESIDENT and RESPONSE are dependent
## chi_sq = 0.7493, df = 3, p_value = 0.8616
```



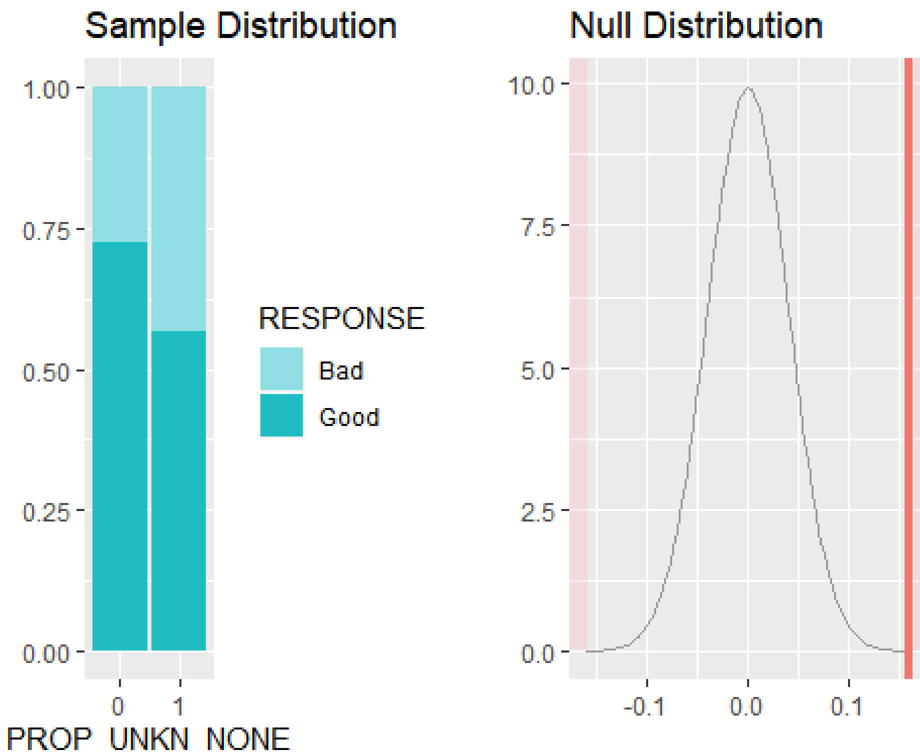

```
inference(data = credit,
          x= REAL_ESTATE,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
```

```
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 718, p_hat_0 = 0.6657
## n_1 = 282, p_hat_1 = 0.7872
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -3.7726
## p_value = 2e-04
```



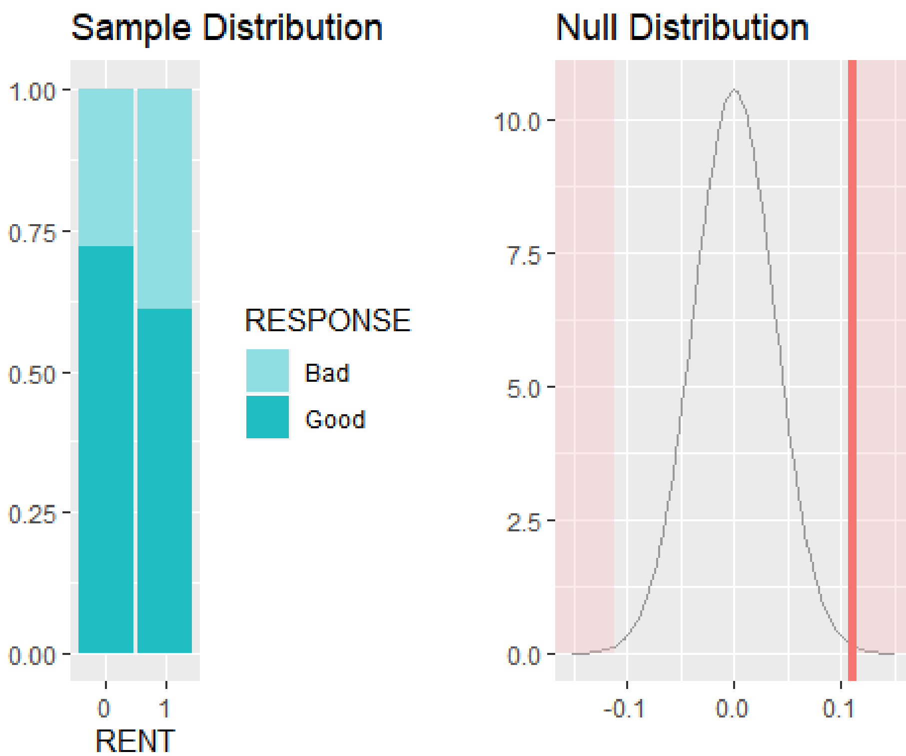

```
inference(data = credit,
          x= PROP_UNKN_NONE,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 846, p_hat_0 = 0.7246
```

```
## n_1 = 154, p_hat_1 = 0.5649
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 3.9766
## p_value = 1e-04
```

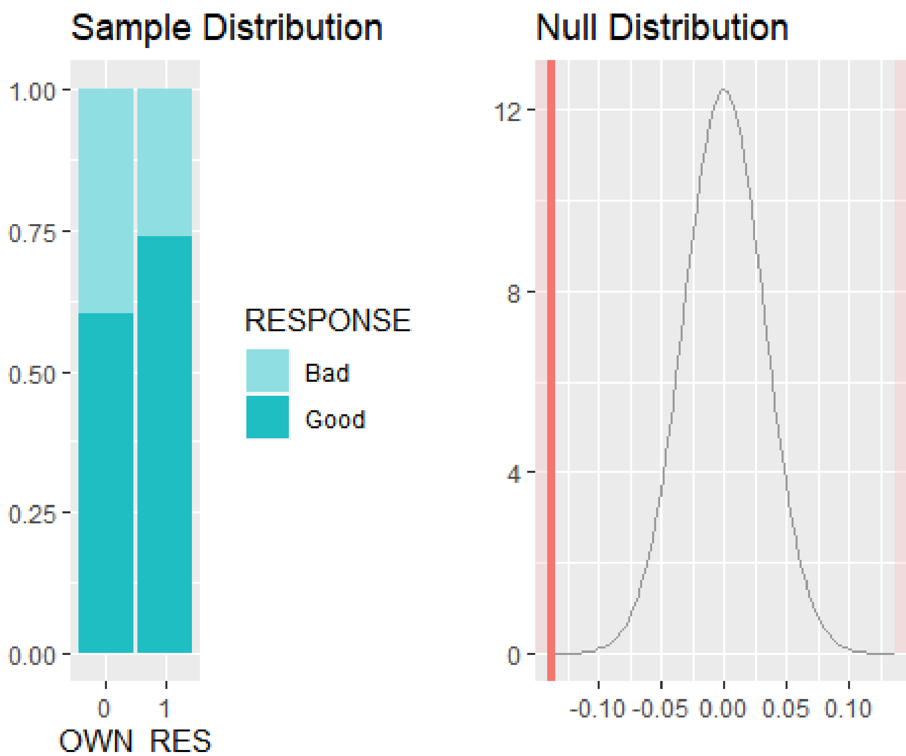



```
inference(data = credit,
          x= RENT,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")
```

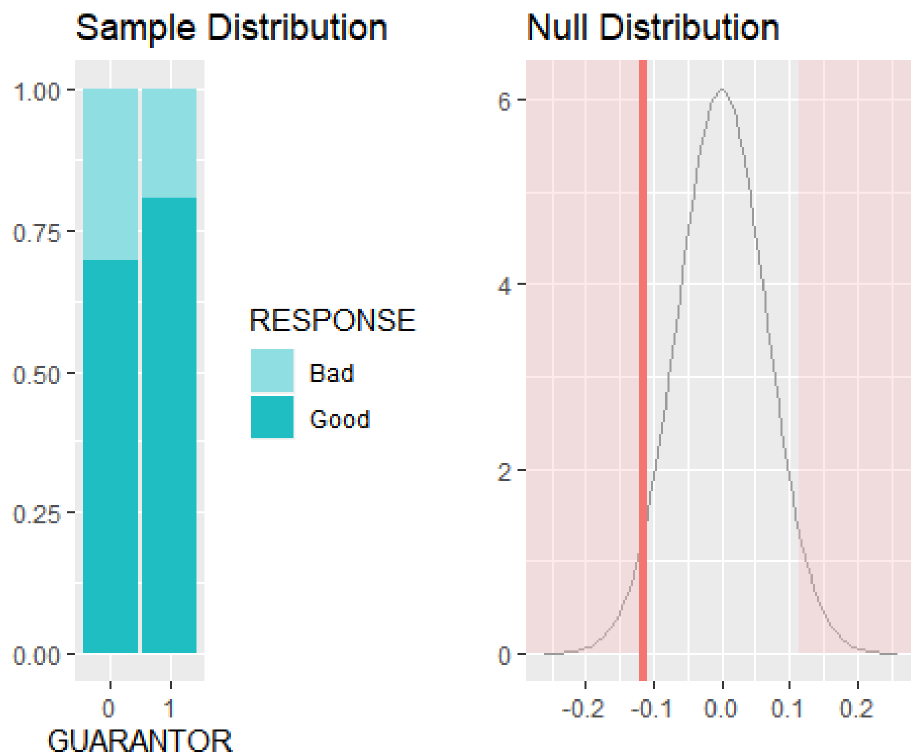
```
## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 821, p_hat_0 = 0.7199
## n_1 = 179, p_hat_1 = 0.6089
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = 2.9341
## p_value = 0.0033
```



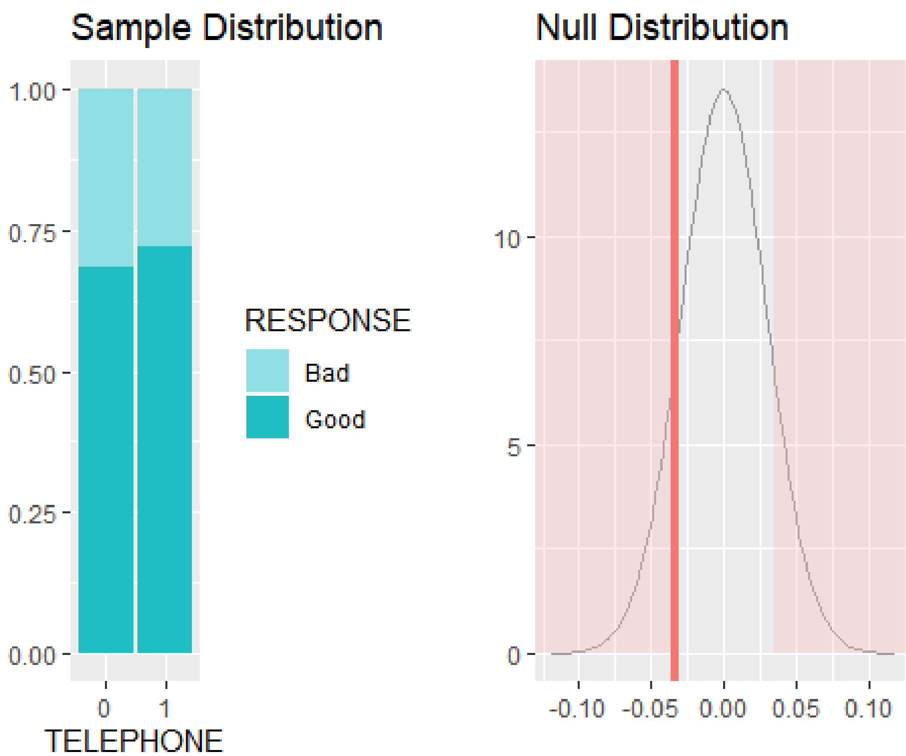

```
inference(data = credit,
          x= OWN_RES,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 287, p_hat_0 = 0.6028
## n_1 = 713, p_hat_1 = 0.7391
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -4.2561
## p_value = < 0.0001
```




```
inference(data = credit,
          x= GUARANTOR,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 948, p_hat_0 = 0.6941
## n_1 = 52, p_hat_1 = 0.8077
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -1.7405
## p_value = 0.0818
```




```
inference(data = credit,
          x= TELEPHONE,
          y=RESPONSE,
          statistic = "proportion",
          type = "ht",
          null = 0,
          success = "Good",
          alternative = "twosided",
          method = "theoretical")

## Response variable: categorical (2 levels, success: Good)
## Explanatory variable: categorical (2 levels)
## n_0 = 596, p_hat_0 = 0.6862
## n_1 = 404, p_hat_1 = 0.7203
## H0: p_0 =  p_1
## HA: p_0 != p_1
## z = -1.1532
## p_value = 0.2488
```




```
glm.fits=glm(RESPONSE~.-RESPONSE,data=credit,family = binomial)
summary(glm.fits)

##
## Call:
## glm(formula = RESPONSE ~ . - RESPONSE, family = binomial, data = cr
edit)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6091  -0.7183   0.3772   0.7050   2.2949
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.786e+00  1.124e+00   1.589 0.112080
## CHK_ACCT1           3.739e-01  2.152e-01   1.737 0.082352 .
## CHK_ACCT2           9.748e-01  3.682e-01   2.647 0.008114 **
## CHK_ACCT3           1.706e+00  2.299e-01   7.421 1.16e-13 ***
## DURATION           -2.793e-02  9.236e-03  -3.024 0.002495 **
## HISTORY1           -7.328e-02  5.424e-01  -0.135 0.892522
## HISTORY2            5.811e-01  4.286e-01   1.356 0.175192
## HISTORY3            8.603e-01  4.696e-01   1.832 0.066914 .
## HISTORY4            1.441e+00  4.385e-01   3.286 0.001016 **
## NEW_CAR1           -8.069e-01  3.873e-01  -2.084 0.037197 *
## USED_CAR1           8.317e-01  4.850e-01   1.715 0.086356 .
```

```
## FURNITURE1       -2.568e-02  4.035e-01  -0.064 0.949249
## Radio_TV1         9.364e-02  3.918e-01   0.239 0.811126
## EDUCATION1       -8.411e-01  5.059e-01  -1.662 0.096416 .
## RETRAINING1      -7.570e-02  4.467e-01  -0.169 0.865415
## AMOUNT           -1.233e-04  4.421e-05  -2.789 0.005283 **
## SAV_ACCT1         3.453e-01  2.856e-01   1.209 0.226589
## SAV_ACCT2         4.030e-01  4.006e-01   1.006 0.314475
## SAV_ACCT3         1.307e+00  5.218e-01   2.504 0.012262 *
## SAV_ACCT4         9.469e-01  2.600e-01   3.642 0.000271 ***
## EMPLOYMENT1       4.558e-02  4.247e-01   0.107 0.914540
## EMPLOYMENT2       1.837e-01  4.087e-01   0.449 0.653160
## EMPLOYMENT3       8.256e-01  4.415e-01   1.870 0.061498 .
## EMPLOYMENT4       2.939e-01  4.103e-01   0.716 0.473828
## INSTALL_RATE     -3.281e-01  8.779e-02  -3.737 0.000186 ***
## MALE_DIV1        -2.742e-01  3.864e-01  -0.710 0.477850
## MALE_SINGLE1      5.372e-01  2.090e-01   2.571 0.010152 *
## MALE_MAR_or_WID1  1.326e-01  3.093e-01   0.429 0.668116
## Coapplicant1     -3.818e-01  4.048e-01  -0.943 0.345616
## GUARANTOR1        9.505e-01  4.188e-01   2.270 0.023225 *
## PRESENT_RESIDENT -3.250e-03  8.611e-02  -0.038 0.969891
## REAL_ESTATE1      2.200e-01  2.141e-01   1.027 0.304231
## PROP_UNKN_NONE1  -5.395e-01  3.841e-01  -1.405 0.160165
## AGE               1.311e-02  9.159e-03   1.431 0.152374
## OTHER_INSTALL1   -5.761e-01  2.123e-01  -2.714 0.006656 **
## RENT1            -7.152e-01  4.713e-01  -1.518 0.129123
## OWN_RES1         -2.753e-01  4.451e-01  -0.619 0.536170
## NUM_CREDITS      -2.835e-01  1.897e-01  -1.495 0.135026
## JOB1             -4.762e-01  6.691e-01  -0.712 0.476720
## JOB2             -5.095e-01  6.461e-01  -0.788 0.430410
## JOB3             -3.865e-01  6.518e-01  -0.593 0.553144
## NUM_DEPENDENTS   -2.612e-01  2.478e-01  -1.054 0.291844
## TELEPHONE1        3.130e-01  2.001e-01   1.564 0.117813
## FOREIGN1          1.396e+00  6.289e-01   2.220 0.026443 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1221.73  on 999  degrees of freedom
## Residual deviance:  899.57  on 956  degrees of freedom
## AIC: 987.57
## ## Number of Fisher Scoring iterations: 5
```

Finding : Following variables have p value MORE than the alpha value (0.05) and these variables do not gave a statistically significant association with RESPONSE:
FURNITURE,

RETRAINING,
MALE_DIV,
MALE_MAR_OR_WID,
PRESENT_RESIDENT,
NUM_CREDITS,
JOB,
NUM_DEPENDENTS,
TELEPHONE,
GUARANTOR

We will remove these variables from our analysis and we remain with 21 variables now.

Problem 5, Part b:

First we divide the data into training and test sets:

```
set.seed(101)
credit <- select(credit,-c(FURNITURE,RETRAINING,MALE_DIV,MALE_MAR_or_W
ID,
                           PRESENT_RESIDENT,NUM_CREDITS,JOB,

                           NUM_DEPENDENTS,TELEPHONE,GUARANTOR))

credit_index <- sample.int(n = nrow(credit), size = floor(.60*nrow(cre
dit)), replace = F)
credittrain <- credit[credit_index,]
credittest <- credit[-credit_index,]
```

We have been given in the problem, that the cost of predicting False positives is 5 times of that predicting False negatives. So we will create a penalty or cost matrix and pass that as a parameter in each of our models.

```
costMatrix <- matrix(c(0,1,5,0), byrow=TRUE, nrow=2)
```

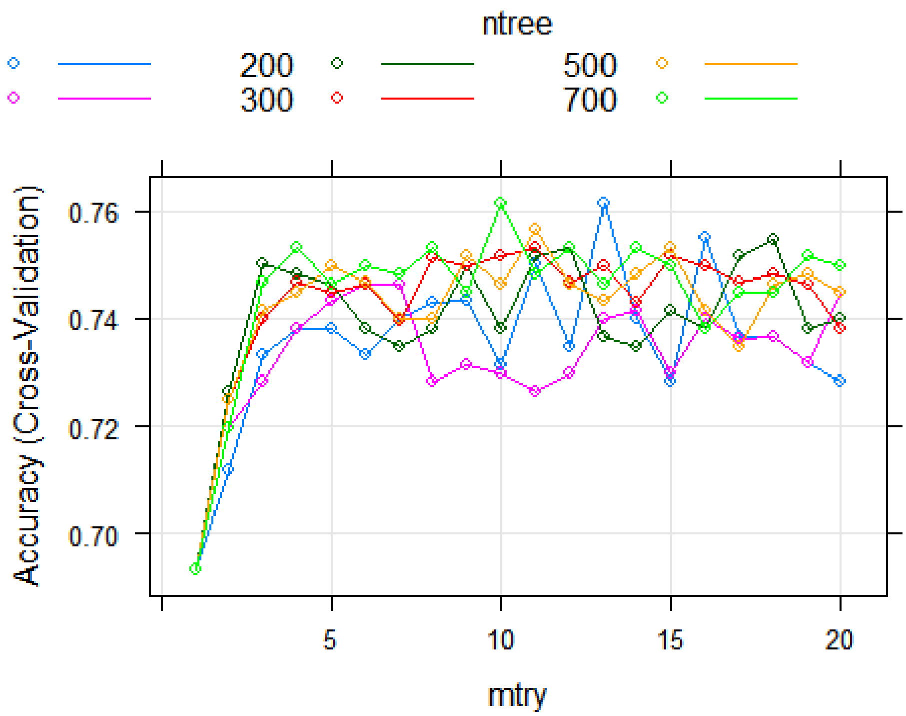
Steps:

1. **Parameter tuning using cross validation. For random Forest, we will find the best values of “mtry” and “ntrees”. For rpart decision tree, we will find the best “cp”.**
2. **We use 10-fold cross validation to compare the model performance of Random Forest and rpart decision tree.**
3. **Construct the new model with fine tuned parameters and predict the classes of test data. Evaluate the performance of both random forest and rpart on the test data.**
4. **Identify the important variables and important output rules for “Good” credit Response.**

```r
# Define the control
control1 <- trainControl(method = "cv",
    number = 10)
```

Part 1.a) Parameter tuning for Random Forest

```r
set.seed(101)

creditRF <- list(type = "Classification", library = "randomForest", lo
op = NULL)
creditRF$parameters <- data.frame(parameter = c("mtry", "ntree"),
                                  class = rep("numeric", 2),
                                  label = c("mtry", "ntree"))

creditRF$grid <- function(x, y, len = NULL, search = "grid") {}
creditRF$fit <- function(x, y, wts, param, lev, last, weights, classPr
obs, ...)
  {
  randomForest(x, y, mtry = param$mtry, ntree=param$ntree, ...)
  }

creditRF$predict <- function(modelFit, newdata, preProc = NULL, submod
els = NULL)
   predict(modelFit, newdata)

creditRF$prob <- function(modelFit, newdata, preProc = NULL, submodels
= NULL)
   predict(modelFit, newdata, type = "prob")

creditRF$sort <- function(x) x[order(x[,1]),]
creditRF$levels <- function(x) x$classes


control <- trainControl(method="cv", number=10)
tunegrid <- expand.grid(.mtry=c(1:20), .ntree=c(50,100,200,300,500, 70
0))
set.seed(111)
creditnew <- train(RESPONSE ~.-RESPONSE,
                   data=credittrain, method=creditRF,
                   metric="Accuracy", tuneGrid=tunegrid,
                   trControl=control1, parms = list(loss=costMatrix))

plot(creditnew)
```




```
creditnew

## 600 samples
##  20 predictor
##   2 classes: 'Bad', 'Good'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 540, 539, 540, 539, 540, 540, ...
## Resampling results across tuning parameters:
##
##   mtry  ntree  Accuracy   Kappa
##    1     50    0.6933546  0.0000000
##    1    100    0.6933546  0.0000000
##    1    200    0.6933546  0.0000000
##    1    300    0.6933546  0.0000000
##    1    500    0.6933546  0.0000000
##    1    700    0.6933546  0.0000000
##    2     50    0.7116343  0.1450165
##    2    100    0.7198574  0.1722191
##    2    200    0.7265796  0.1928608
##    2    300    0.7249139  0.1816718
##    2    500    0.7248856  0.1771838
##    2    700    0.7198856  0.1621148
##    3     50    0.7333028  0.2795321
```

```
##    3    100    0.7281643  0.2574198
##    3    200    0.7499995  0.3126147
##    3    300    0.7398884  0.2740638
##    3    500    0.7415824  0.2785174
##    3    700    0.7465824  0.2913710
##    4     50    0.7381643  0.3001947
##    4    100    0.7383055  0.3006459
##    4    200    0.7483630  0.3175287
##    4    300    0.7466106  0.3129757
##    4    500    0.7449166  0.3058894
##    4    700    0.7531944  0.3383718
##    5     50    0.7382472  0.3054987
##    5    100    0.7432773  0.3156976
##    5    200    0.7465560  0.3250346
##    5    300    0.7448046  0.3081825
##    5    500    0.7499176  0.3289985
##    5    700    0.7465277  0.3225225
##    6     50    0.7330805  0.3038361
##    6    100    0.7465569  0.3308168
##    6    200    0.7381379  0.3004286
##    6    300    0.7465551  0.3238779
##    6    500    0.7465824  0.3228237
##    6    700    0.7499166  0.3345468
##    7     50    0.7400824  0.3311836
##    7    100    0.7465560  0.3276668
##    7    200    0.7348865  0.2950334
##    7    300    0.7397490  0.3124973
##    7    500    0.7398592  0.3115584
##    7    700    0.7482227  0.3346526
##    8     50    0.7431388  0.3272841
##    8    100    0.7282491  0.2855038
##    8    200    0.7382208  0.3068897
##    8    300    0.7514157  0.3452604
##    8    500    0.7398875  0.3126659
##    8    700    0.7531944  0.3389953
##    9     50    0.7433338  0.3374023
##    9    100    0.7314439  0.2997171
##    9    200    0.7499440  0.3360318
##    9    300    0.7498884  0.3439979
##    9    500    0.7516389  0.3452840
##    9    700    0.7449440  0.3226347
##   10     50    0.7315259  0.2996093
##   10    100    0.7298310  0.3110921
##   10    200    0.7382491  0.3119907
##   10    300    0.7515842  0.3476755
##   10    500    0.7465277  0.3371718
```

```
## 10 700 0.7614731 0.3714292
## 11  50 0.7499722 0.3573550
## 11 100 0.7265541 0.2960287
## 11 200 0.7515004 0.3605953
## 11 300 0.7533338 0.3557465
## 11 500 0.7566389 0.3642256
## 11 700 0.7482227 0.3421750
## 12  50 0.7348291 0.3184963
## 12 100 0.7299704 0.3116487
## 12 200 0.7531935 0.3581323
## 12 300 0.7465824 0.3403233
## 12 500 0.7464712 0.3418150
## 12 700 0.7532217 0.3556020
## 13  50 0.7615842 0.3853153
## 13 100 0.7398328 0.3294253
## 13 200 0.7364712 0.3202231
## 13 300 0.7498055 0.3514810
## 13 500 0.7432217 0.3342423
## 13 700 0.7464995 0.3384613
## 14  50 0.7399412 0.3247673
## 14 100 0.7415532 0.3265088
## 14 200 0.7348036 0.3191534
## 14 300 0.7431935 0.3377276
## 14 500 0.7481379 0.3541300
## 14 700 0.7532236 0.3638294
## 15  50 0.7282199 0.3110893
## 15 100 0.7298300 0.3105987
## 15 200 0.7415268 0.3392324
## 15 300 0.7515541 0.3646816
## 15 500 0.7532227 0.3664088
## 15 700 0.7498319 0.3533391
## 16  50 0.7549176 0.3742967
## 16 100 0.7398865 0.3442390
## 16 200 0.7381379 0.3260184
## 16 300 0.7498046 0.3573430
## 16 500 0.7413601 0.3383903
## 16 700 0.7381643 0.3314839
## 17  50 0.7365824 0.3351084
## 17 100 0.7364147 0.3317485
## 17 200 0.7515541 0.3667418
## 17 300 0.7466088 0.3482245
## 17 500 0.7348310 0.3279788
## 17 700 0.7448319 0.3433742
## 18  50 0.7364448 0.3304556
## 18 100 0.7364722 0.3353238
## 18 200 0.7547518 0.3735610
```

```
##    18     300     0.7482217  0.3542758
##    18     500     0.7465551  0.3585331
##    18     700     0.7448893  0.3504925
##    19      50     0.7315815  0.3253837
##    19     100     0.7315824  0.3241653
##    19     200     0.7382217  0.3358702
##    19     300     0.7465541  0.3530014
##    19     500     0.7483046  0.3557964
##    19     700     0.7515560  0.3649345
##    20      50     0.7281652  0.3232142
##    20     100     0.7449449  0.3519525
##    20     200     0.7398865  0.3391991
##    20     300     0.7381926  0.3340243
##    20     500     0.7448601  0.3545949
##    20     700     0.7499166  0.3642619
## 
## Accuracy was used to select the optimal model using the largest val
ue.
## The final values used for the model were mtry = 13 and ntree = 50.
```

From the results of parameter tuning by cross validation, the final values used for the random forest model were mtry = 13 and ntree = 50.

Part 1.b) Parameter tuning for rpart Decision tree

```
defaultrpart <- rpart(RESPONSE~.-RESPONSE,
                      data = credittrain, method = "class",
                      control = rpart.control(minsplit = 10, cp = 0.01
),
                      parms = list(loss=costMatrix))

printcp(defaultrpart)

## 
## Classification tree:
## rpart(formula = RESPONSE ~ . - RESPONSE, data = credittrain, 
##     method = "class", parms = list(loss = costMatrix), control = rp
art.control(minsplit = 10, 
##         cp = 0.01))
## 
## Variables actually used in tree construction:
##  [1] AGE         AMOUNT      CHK_ACCT    DURATION    EDUCATION  
##  [6] EMPLOYMENT  HISTORY     NEW_CAR     REAL_ESTATE SAV_ACCT   
## 
## Root node error: 184/600 = 0.30667
## 
## n= 600
```

```
##
##          CP nsplit rel error xerror    xstd
## 1 0.027174      0   1.00000 5.0000 0.30692
## 2 0.021739      3   0.91848 4.7826 0.30207
## 3 0.016304      5   0.87500 4.4674 0.29448
## 4 0.013975      8   0.82609 4.5054 0.29386
## 5 0.013587     18   0.67935 4.4620 0.29255
## 6 0.010000     20   0.65217 4.2554 0.28661
```

From the results of parameter tuning by cross validation, we see that the xerror is minimum for complexity parameter value cp = 0.01 and minsplit = 20.

Part 2.a) 10-Fold Cross Validation on Random Forest

```
set.seed(112)

k=10
nc = floor(nrow(credit)/k)
acc.vect = rep(NA,k)

for (i in 1:k) {

  c1 = ((i-1) * nc+1)
  c2 = (i*nc)
  subset = c1:c2

  cvc.train = credit[-subset,]
  cvc.test = credit[subset,]

  creditrf <- randomForest (RESPONSE~.-RESPONSE,
                            data = cvc.train, mtry = 13, ntree = 50,
                            parms = list(loss=costMatrix))

  creditpred <- predict(creditrf, newdata = cvc.test, type = "class")

  acc.vect[i] <-  (confusionMatrix(creditpred, cvc.test$RESPONSE)$over
all)[1]

  ##acc.vect[i] <-  auc(roc(cvc.test[,12], creditpred[,2]))

  print(paste("Accuracy for fold", i, ":", acc.vect[i]))

}

## [1] "Accuracy for fold 1 : 0.81"
## [1] "Accuracy for fold 2 : 0.7"
## [1] "Accuracy for fold 3 : 0.76"
```

```
## [1] "Accuracy for fold 4 : 0.75"
## [1] "Accuracy for fold 5 : 0.72"
## [1] "Accuracy for fold 6 : 0.7"
## [1] "Accuracy for fold 7 : 0.77"
## [1] "Accuracy for fold 8 : 0.79"
## [1] "Accuracy for fold 9 : 0.79"
## [1] "Accuracy for fold 10 : 0.7"

print(paste("Average Accuracy :", mean(acc.vect)))

## [1] "Average Accuracy : 0.749"
```

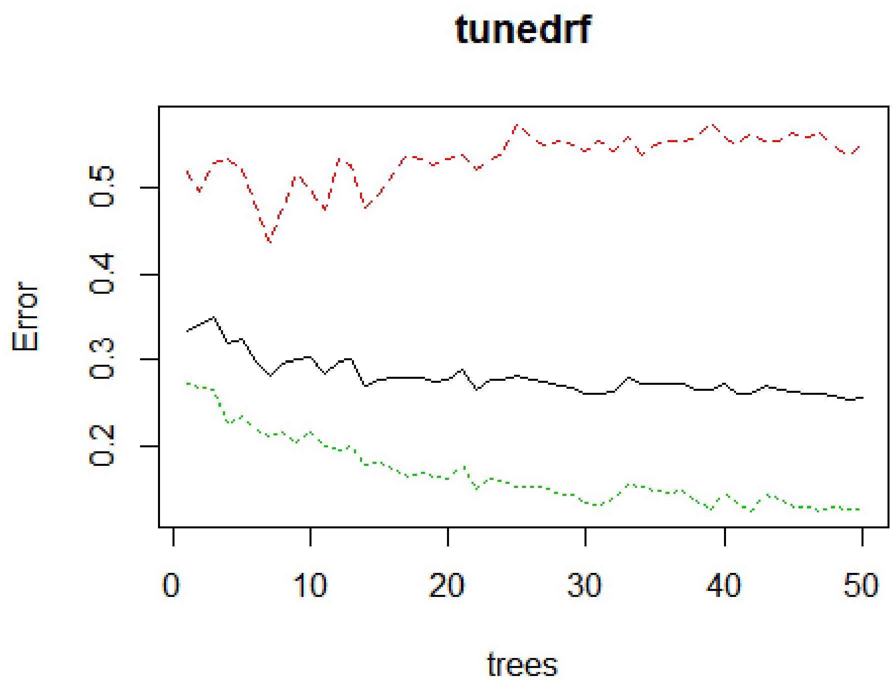
Part 2.b) 10-Fold Cross Validation on rpart decision tree

```
set.seed(122)

k2=10
n1 = floor(nrow(credit)/k2)
err.vector = rep(NA,k)

for (i in 1:k2) {

  p1 = ((i-1) * n1+1)
  p2 = (i*n1)
  credsubset = p1:p2

  cvrpart.train = credit[-credsubset, ]
  cvrpart.test = credit[credsubset,]

  creditrpart <- rpart(RESPONSE ~ .-RESPONSE,
                       data = cvrpart.train, method = "class", cp = 0.
01,
                       minsplit = 20,
                       parms = list(loss=costMatrix))


  creditpred_rpart <- predict(creditrpart, newdata = cvrpart.test, typ
e = "class")


  err.vector[i] <- (confusionMatrix(creditpred_rpart, cvrpart.test$RES
PONSE)$overall)[1]

  print(paste("Accuracy for fold", i, ":", err.vector[i]))
```

```
}
## [1] "Accuracy for fold 1 : 0.78"
## [1] "Accuracy for fold 2 : 0.69"
## [1] "Accuracy for fold 3 : 0.8"
## [1] "Accuracy for fold 4 : 0.73"
## [1] "Accuracy for fold 5 : 0.74"
## [1] "Accuracy for fold 6 : 0.6"
## [1] "Accuracy for fold 7 : 0.67"
## [1] "Accuracy for fold 8 : 0.68"
## [1] "Accuracy for fold 9 : 0.72"
## [1] "Accuracy for fold 10 : 0.69"

print(paste("Average Accuracy :", mean(err.vector)))

## [1] "Average Accuracy : 0.71"
```

Part 3.a) Passing the parameters received from paramenter tuning and creating a more fine-tuned random forest tree. Also we predict the "RESPONSE" class for our test data with the tuned random forest model. We will also find out the area under the ROC curve for our model.

```
tunedrf <- (randomForest(RESPONSE ~ .-RESPONSE,
                         data = credittrain, replace = TRUE,
                         proximity = TRUE, importance = TRUE, mtry = 13
, ntree = 50,
                         parms = list(loss=costMatrix), cutoff = c(0.5,
0.5)))

plot(tunedrf)
```




```
rf_pred <- predict(tunedrf, newdata= credittest, type = "prob", positi
ve = "Good")

confusionMatrix(predict(tunedrf, newdata= credittest, type = "class"),
                credittest$RESPONSE, positive='Good')

## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad   40   21
##       Good  76  263
##
##                Accuracy : 0.7575
##                  95% CI : (0.7124, 0.7987)
##     No Information Rate : 0.71
##     P-Value [Acc > NIR] : 0.01944
##
##                   Kappa : 0.3151
##
##  Mcnemar's Test P-Value : 4.185e-08
##
##             Sensitivity : 0.9261
##             Specificity : 0.3448
##          Pos Pred Value : 0.7758
```

```
##          Neg Pred Value : 0.6557
##              Prevalence : 0.7100
##          Detection Rate : 0.6575
##    Detection Prevalence : 0.8475
##       Balanced Accuracy : 0.6354
##
##        'Positive' Class : Good
##
```

```
rfauc <- prediction(rf_pred[,2], credittest$RESPONSE)
```

```
performance(rfauc,"auc")
```

```
## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.765238
##
##
## Slot "alpha.values":
## list()
```

Results from Tuned Random Forest tree on test data:

1. Accuracy: 75.75%
2. False Positive rate: (1-specificity) = 65.52%
3. False Negative: (1 - Sensitivity) x Prevalence = 5.2%
4. Recall: sensitivity = 92.61%
5. AUC: 0.765

Part 3.b) Passing the parameters received from paramenter tuning and creating a more fine-tuned rpart tree. Also we predict the "RESPONSE" class for our test data with the tuned rpart model. We will also find out the area under the ROC curve for our model.

```
opt = which.min(defaultrpart$cptable[,"xerror"])
  cp = defaultrpart$cptable[opt, "CP"]
  prunedrpart = prune(defaultrpart, cp = cp, minsplit = 20)

  prunedrpart
```

```
## n= 600
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 600 184 Good (0.30666667 0.69333333)
##     2) CHK_ACCT=0,1 343 154 Good (0.44897959 0.55102041)
##       4) DURATION>=43.5 36  29 Good (0.80555556 0.19444444)
##         8) SAV_ACCT=0,3 27  10 Bad (0.92592593 0.07407407)
##          16) EMPLOYMENT=2,4 17   0 Bad (1.00000000 0.00000000) *
##          17) EMPLOYMENT=1,3 10   8 Good (0.80000000 0.20000000)
##            34) AGE< 34.5 6   0 Bad (1.00000000 0.00000000) *
##            35) AGE>=34.5 4   2 Good (0.50000000 0.50000000) *
##         9) SAV_ACCT=1,4 9   4 Good (0.44444444 0.55555556) *
##       5) DURATION< 43.5 307 125 Good (0.40716612 0.59283388)
##        10) HISTORY=0,1 34  23 Good (0.67647059 0.32352941)
##          20) NEW_CAR=1 12   5 Bad (0.91666667 0.08333333)
##            40) DURATION< 19.5 9   0 Bad (1.00000000 0.00000000) *
##            41) DURATION>=19.5 3   2 Good (0.66666667 0.33333333) *
##          21) NEW_CAR=0 22  12 Good (0.54545455 0.45454545) *
##        11) HISTORY=2,3,4 273 102 Good (0.37362637 0.62637363)
##          22) AMOUNT>=8015.5 18  13 Good (0.72222222 0.27777778)
##            44) AMOUNT< 8428.5 5   0 Bad (1.00000000 0.00000000) *
##            45) AMOUNT>=8428.5 13   8 Good (0.61538462 0.38461538)
##              90) EMPLOYMENT=1 3   0 Bad (1.00000000 0.00000000) *
##              91) EMPLOYMENT=0,2,3,4 10   5 Good (0.50000000 0.50000
000) *
##          23) AMOUNT< 8015.5 255  89 Good (0.34901961 0.65098039)
##            46) AMOUNT< 1285 64  33 Good (0.51562500 0.48437500)
##              92) AMOUNT>=1206.5 11   5 Bad (0.90909091 0.09090909)
##               184) AGE< 35.5 8   0 Bad (1.00000000 0.00000000) *
##               185) AGE>=35.5 3   2 Good (0.66666667 0.33333333) *
##              93) AMOUNT< 1206.5 53  23 Good (0.43396226 0.56603774)
##               186) EDUCATION=1 3   0 Bad (1.00000000 0.00000000) *
##               187) EDUCATION=0 50  20 Good (0.40000000 0.60000000)
##                 374) REAL_ESTATE=0 28  17 Good (0.60714286 0.392857
14)
##                   748) SAV_ACCT=1,4 5   0 Bad (1.00000000 0.0000000
0) *
##                   749) SAV_ACCT=0 23  12 Good (0.52173913 0.4782608
```

```
7) *
##                   375) REAL_ESTATE=1 22   3 Good (0.13636364 0.863636
36) *
##              47) AMOUNT>=1285 191  56 Good (0.29319372 0.70680628)
##                94) AGE< 25.5 48  22 Good (0.45833333 0.54166667)
##                 188) DURATION>=31.5 4   0 Bad (1.00000000 0.00000000)
*
##                 189) DURATION< 31.5 44  18 Good (0.40909091 0.5909090
9)
##                   378) AMOUNT< 1491 4   0 Bad (1.00000000 0.00000000)
*
##                   379) AMOUNT>=1491 40  14 Good (0.35000000 0.6500000
0) *
##                95) AGE>=25.5 143  34 Good (0.23776224 0.76223776) *
##     3) CHK_ACCT=2,3 257  30 Good (0.11673152 0.88326848) *
```

```
pred_rpart <- predict(prunedrpart, credittest, type = "prob", positive
="Good")


confusionMatrix(credittest$RESPONSE,
                predict(prunedrpart, credittest, type = "class"),
                positive = "Good")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad   13  103
##       Good  13  271
##
##                Accuracy : 0.71
##                  95% CI : (0.6628, 0.754)
##     No Information Rate : 0.935
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.086
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7246
##             Specificity : 0.5000
##          Pos Pred Value : 0.9542
##          Neg Pred Value : 0.1121
##              Prevalence : 0.9350
##          Detection Rate : 0.6775
```

```
##    Detection Prevalence : 0.7100
##       Balanced Accuracy : 0.6123
##
##        'Positive' Class : Good
##
```

```
rpartauc <- prediction(pred_rpart[,2], credittest$RESPONSE)

performance(rpartauc,"auc")
```

```
## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.7108578
##
##
## Slot "alpha.values":
## list()
```

Results from Tuned rpart tree on the test data:

1. Accuracy: 71%

2. False Positive rate: (1-specificity) = 50%

3. False Negative: (1 - Sensitivity) x Prevalence = 25.74%

4. Recall: sensitivity = 72.46%

5. AUC: 0.71

Important Variables from both the models

Part 4.a) To identify the best nodes in the random forest, we use the 'importance' parameter.

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

```
##                    MeanDecreaseGini
## CHK_ACCT                 33.4220177
## DURATION                 29.8124965
## HISTORY                  15.9914626
```

```
## NEW_CAR                    4.5327474
## USED_CAR                   1.4510618
## FURNITURE                  2.3757948
## Radio_TV                   2.7750862
## EDUCATION                  1.8718100
## RETRAINING                 3.0195923
## AMOUNT                    39.5346768
## SAV_ACCT                  11.4983449
## EMPLOYMENT                15.8517103
## INSTALL_RATE               9.1652626
## MALE_DIV                   2.0041005
## MALE_SINGLE                3.2041263
## MALE_MAR_or_WID            2.1522087
## Coapplicant                2.2848437
## GUARANTOR                  2.9485227
## PRESENT_RESIDENT           9.7283722
## REAL_ESTATE                3.9370521
## PROP_UNKN_NONE             2.9909234
## AGE                       27.3956251
## OTHER_INSTALL              3.7394541
## RENT                       2.5269606
## OWN_RES                    2.6097297
## NUM_CREDITS                4.5668098
## JOB                        7.0685160
## NUM_DEPENDENTS             2.9000351
## TELEPHONE                  2.3422806
## FOREIGN                    0.7065976
```

The most important variables in the random forest are the ones with highest value of MeanDecreaseGini. So according to above results, "AMOUNT", "CHK_ACCT","AGE" "DURATION", "EMPLOYEMENT", "HISTORY" and "INSTALL_RATE" seem to be the most important variables.

Part 4.b) To identify the best nodes in the rpart, we extract the variable.importance column from our prunedrpart model and we use asRules() function to exctract the important rules.

```
library(rattle)

## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
prunedrpart$variable.importance
```

```
##        AMOUNT      DURATION      SAV_ACCT      CHK_ACCT           A
GE
##     8.4579110     7.0095536     6.9934774     4.7418654     3.41764
04
##    EMPLOYMENT       HISTORY       NEW_CAR     EDUCATION      Radio_
TV
##     3.0546321     2.5364646     2.3833041     1.5530345     1.10940
00
##   REAL_ESTATE       OWN_RES  INSTALL_RATE      USED_CAR   Coapplica
nt
##     0.9290377     0.7361602     0.4290762     0.3680801     0.35493
44
##   MALE_SINGLE OTHER_INSTALL
##     0.3365304     0.2741192
```

```
asRules(prunedrpart)
```

```
##
##  Rule number: 3 [RESPONSE=Good cover=257 (43%) prob=0.88]
##    CHK_ACCT=2,3
##
##  Rule number: 375 [RESPONSE=Good cover=22 (4%) prob=0.86]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT< 1285
##    AMOUNT< 1206
##    EDUCATION=0
##    REAL_ESTATE=1
##
##  Rule number: 95 [RESPONSE=Good cover=143 (24%) prob=0.76]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT>=1285
##    AGE>=25.5
##
##  Rule number: 379 [RESPONSE=Good cover=40 (7%) prob=0.65]
##    CHK_ACCT=0,1
```

```
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT>=1285
##    AGE< 25.5
##    DURATION< 31.5
##    AMOUNT>=1491
##
##  Rule number: 9 [RESPONSE=Good cover=9 (2%) prob=0.56]
##    CHK_ACCT=0,1
##    DURATION>=43.5
##    SAV_ACCT=1,4
##
##  Rule number: 35 [RESPONSE=Good cover=4 (1%) prob=0.50]
##    CHK_ACCT=0,1
##    DURATION>=43.5
##    SAV_ACCT=0,3
##    EMPLOYMENT=1,3
##    AGE>=34.5
##
##  Rule number: 91 [RESPONSE=Good cover=10 (2%) prob=0.50]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT>=8016
##    AMOUNT>=8428
##    EMPLOYMENT=0,2,3,4
##
##  Rule number: 749 [RESPONSE=Good cover=23 (4%) prob=0.48]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT< 1285
##    AMOUNT< 1206
##    EDUCATION=0
##    REAL_ESTATE=0
##    SAV_ACCT=0
##
##  Rule number: 21 [RESPONSE=Good cover=22 (4%) prob=0.45]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=0,1
##    NEW_CAR=0
##
##  Rule number: 185 [RESPONSE=Good cover=3 (0%) prob=0.33]
```

```
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT< 1285
##    AMOUNT>=1206
##    AGE>=35.5
##
##  Rule number: 41 [RESPONSE=Good cover=3 (0%) prob=0.33]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=0,1
##    NEW_CAR=1
##    DURATION>=19.5
##
##  Rule number: 378 [RESPONSE=Bad cover=4 (1%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT>=1285
##    AGE< 25.5
##    DURATION< 31.5
##    AMOUNT< 1491
##
##  Rule number: 188 [RESPONSE=Bad cover=4 (1%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT>=1285
##    AGE< 25.5
##    DURATION>=31.5
##
##  Rule number: 748 [RESPONSE=Bad cover=5 (1%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT< 1285
##    AMOUNT< 1206
##    EDUCATION=0
##    REAL_ESTATE=0
##    SAV_ACCT=1,4
##
##  Rule number: 186 [RESPONSE=Bad cover=3 (0%) prob=0.00]
```

```
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT< 1285
##    AMOUNT< 1206
##    EDUCATION=1
##
##  Rule number: 184 [RESPONSE=Bad cover=8 (1%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT< 8016
##    AMOUNT< 1285
##    AMOUNT>=1206
##    AGE< 35.5
##
##  Rule number: 90 [RESPONSE=Bad cover=3 (0%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT>=8016
##    AMOUNT>=8428
##    EMPLOYMENT=1
##
##  Rule number: 44 [RESPONSE=Bad cover=5 (1%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=2,3,4
##    AMOUNT>=8016
##    AMOUNT< 8428
##
##  Rule number: 40 [RESPONSE=Bad cover=9 (2%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION< 43.5
##    HISTORY=0,1
##    NEW_CAR=1
##    DURATION< 19.5
##
##  Rule number: 34 [RESPONSE=Bad cover=6 (1%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION>=43.5
##    SAV_ACCT=0,3
##    EMPLOYMENT=1,3
##    AGE< 34.5
##
```

```
##  Rule number: 16 [RESPONSE=Bad cover=17 (3%) prob=0.00]
##    CHK_ACCT=0,1
##    DURATION>=43.5
##    SAV_ACCT=0,3
##    EMPLOYMENT=2,4
```

The most important nodes in rpart decision tree for predicting "Good" response are, in order, "CHK_ACCT", "DURATION", "AMOUNT", "HISTORY" and "AGE", because the rules with these variables have high confidence and high support.

Problem 5, Part c:

We have been given that the cost of False positive is 5 times the cost of false negative. We will look for the model having a fair balance between false positive rate and false negative rate. The rpart decision tree has a false positive rate of 0.50 as compared to random forest which has a rate of 0.65.

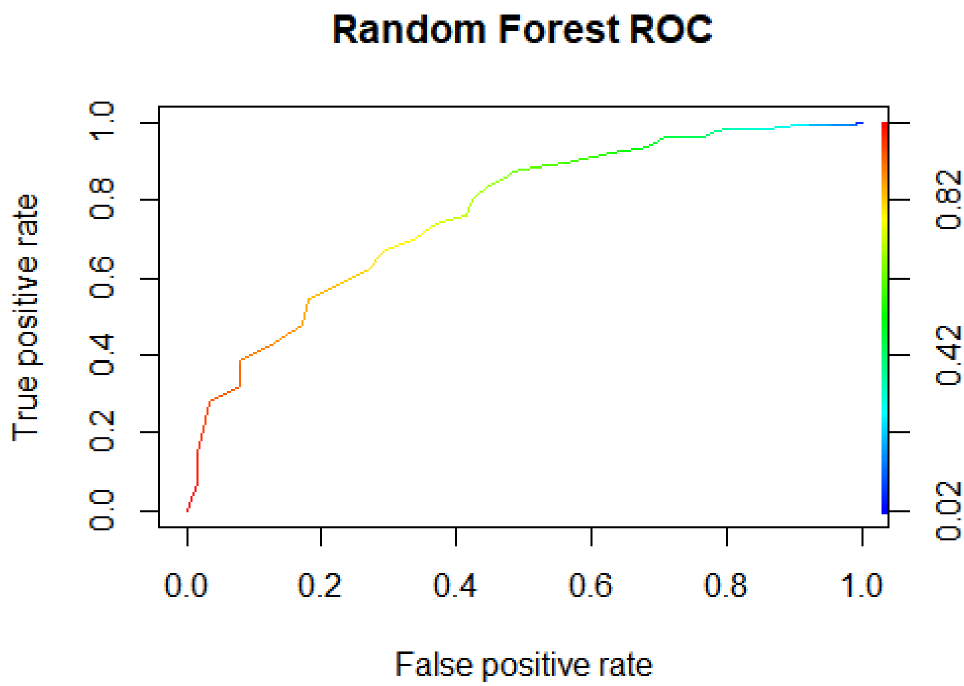
So, the cost of choosing random forest model comes out to be :
((400)+(21100)+(76500)+(2630)) = DM 40,100
Cost of choosing rpart model comes out to be :
((130)+(103100)+(13500)+(2710)) = DM 16,800

So the rpart decision tree reduces the cost. But the false negative rate for rpart is 25.32% as compared to random forest that has false negative rate of 5.2%.

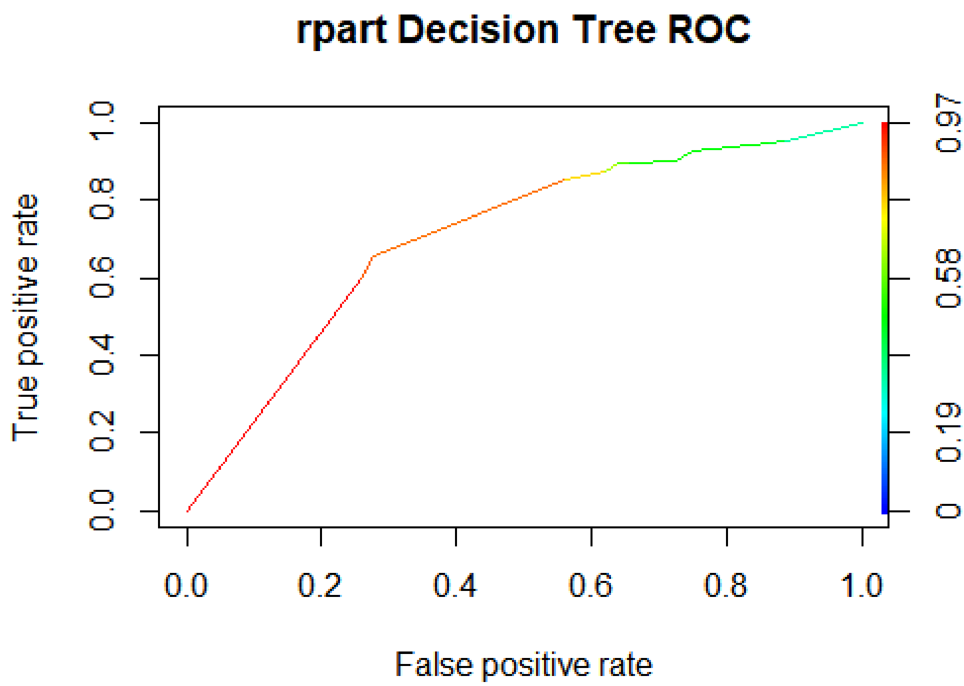
So we see that there is a trade-off between false positive rate and false negative rate. We have to find a model which takes care of this trade-off and provides a optimal and sensible result. For this purpose we will look at the receiver operating curve (ROC). ROC does not actually control False Positive and False negative rates but it tries to find a balance between them.

The Area under the ROC curve to compare the two models, 'tunedrf' and 'prunedrpart'.

```
rfauc <- prediction(rf_pred[,2], credittest$RESPONSE)

ROCrf <- performance(rfauc,"tpr", "fpr")
plot(ROCrf, colorize = TRUE, main = "Random Forest ROC")
```




```
rpartauc <- prediction(pred_rpart[,2], credittest$RESPONSE)

ROCrpart <- performance(rpartauc,"tpr", "fpr")
plot(ROCrpart, colorize = TRUE, main = "rpart Decision Tree ROC")
```




```
performance(rpartauc,"auc")

## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.7108578
##
##
## Slot "alpha.values":
## list()
```

```
performance(rfauc,"auc")

## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.765238
##
##
## Slot "alpha.values":
## list()
```

Finding:
So, according to Area under the curve for model evaluation, we will choose the rpart decision tree. AUC for Random Forest model: 0.71
AUC for rpart model: 0.765

Problem 5, Part d:

By intuition, if we want to penalize the false positives more than false negatives, we should increase the threshold or cut-off points, as that would discourage picking positive class. But to find the optimal cut-off point, we can use the following function:

To find the best cut-off, we will use the Youden Index that gives equal importance to sensitivity and specificity. Youden Index is given by:
Youden index = sensitivity + specificity - 1

We will try to maximize this index and find the corresponding “tpr” - y-intercept, “false_alarm” - x-intercept and “cutoff” - alpha-value for the chosen point.

```
opt.cut <- function(ROCrf){

  cut.ind <- mapply(FUN = function(x,y,p){yi=(y+(1-x)-1)
```

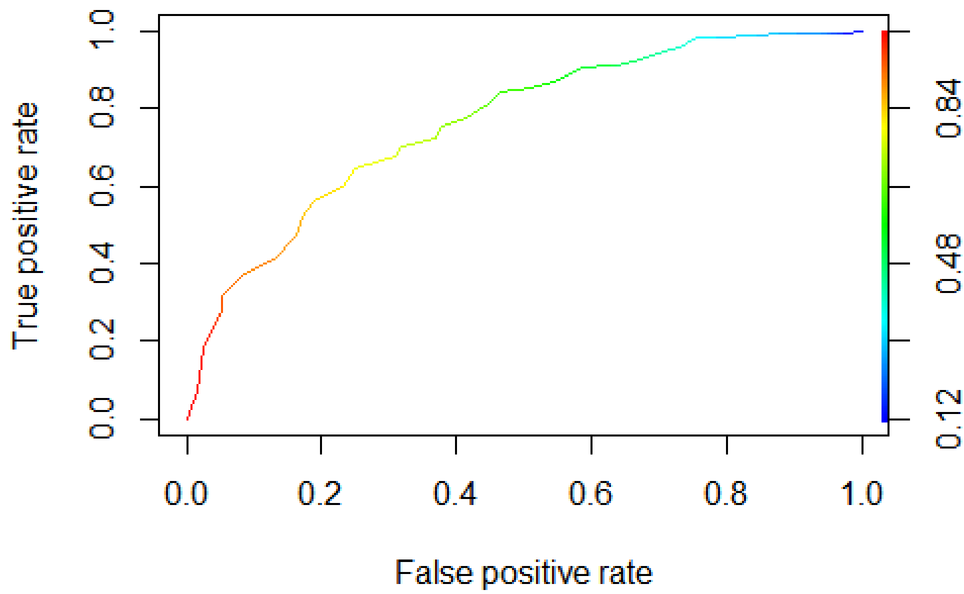
```
  ind<- which(yi==max(yi))
  c(recall = y[[ind]], specificity = 1-x[[ind]],cutoff = p[[ind]])},RO
Crf@x.values, ROCrf@y.values,ROCrf@alpha.values)
}

print(opt.cut(ROCrf))

##                    [,1]
## recall      0.8732394
## specificity 0.5172414
## cutoff      0.6200000
```

The results show the optimum values of tpr, fpr and threshold for our model, which are:

1. Recall (tpr): 0.87
2. False alarm (fpr): (1-0.51) = 0.49
3. Threshold: (0.62,0.38)

We construct the new model using cut-offs obtained in the previous step:

```
newrf <- (randomForest(RESPONSE ~ .-RESPONSE,
                        data = credittrain, replace = TRUE,
                        proximity = TRUE, importance = TRUE, mtry = 13
,
                        ntree = 50, cutoff = c(0.62,0.38),
                        parms = list(loss=costMatrix)))

rf_prednew <- predict(newrf, newdata= credittest, type = "prob", posit
ive = "Good")

rfaucnew <- prediction(rf_prednew[,2], credittest$RESPONSE)

ROCnew <- performance(rfaucnew,"tpr", "fpr")
plot(ROCnew, colorize = TRUE)
```




```
performance(rfaucnew,"auc")

## An object of class "performance"
## Slot "x.name":
## [1] "None"
##
## Slot "y.name":
## [1] "Area under the ROC curve"
##
## Slot "alpha.name":
## [1] "none"
##
## Slot "x.values":
## list()
##
## Slot "y.values":
## [[1]]
## [1] 0.762673
##
```

```
##
## Slot "alpha.values":
## list()
```

```r
confusionMatrix(predict(newrf, credittest, type = "class"),
                credittest$RESPONSE, positive = "Good" )
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Bad Good
##       Bad   31   13
##       Good  85  271
##
##                Accuracy : 0.755
##                  95% CI : (0.7098, 0.7964)
##     No Information Rate : 0.71
##     P-Value [Acc > NIR] : 0.0255
##
##                   Kappa : 0.2713
##
##  Mcnemar's Test P-Value : 7.387e-13
##
##             Sensitivity : 0.9542
##             Specificity : 0.2672
##          Pos Pred Value : 0.7612
##          Neg Pred Value : 0.7045
##              Prevalence : 0.7100
##          Detection Rate : 0.6775
##    Detection Prevalence : 0.8900
##       Balanced Accuracy : 0.6107
##
##        'Positive' Class : Good
##
```

The measures of the final model on the test data are:
1. AUC: 0.76
2. Accuracy: 75.5

Problem 5, Part e:

Following is a summary of our solution:

1. We start by converting the categorical variables into factors and get a summary of the data-set.

2. Parameter tuning using cross validation for each tree:

    - Random forest: mtry = 13, ntree = 50

- rpart: cp = 0.01, minsplit = 20

3. We use 10-fold cross validation for model evaluation for Random forest and rpart decision tree on the basis of accuracy:
   - Random forest average accuracy for 10 folds: 74.9%
   - rpart average accuracy for 10 folds: 71%
4. Important variables and output rules for “Good” response prediction:
   - Random Forest: “AMOUNT”, “CHK_ACCT”,“AGE” “DURATION”, “EMPLOYEMENT”, “HISTORY” and “INSTALL_RATE”
   - rpart: “CHK_ACCT”, “DURATION”, “AMOUNT”, “HISTORY” and “AGE”
5. Comparing the AUC of two models to determine which is better for our problem. AUC of random forest is higher, so we will choose random forest.
   - AUC for Random Forest model: 0.76
   - AUC for rpart model: 0.71
6. To penalize the false positives and strike a balance between false positive and false negagtive rate, we will determine the best cut-off point on the ROC of random forest using Youden Index. We find (0.62,0.38) as the ideal cut-off point.
7. We train the model on the original training data using the best parameters and the best cut-off and test the model on Test data. The final performance on test data comes out as:
   - AUC: 0.76
   - Accuracy: 75.5%
   - Recall: .95
   - Specificity: 0.26