# Driving Range Prediction

## R Markdown

In this model, we will try and predict the “Trip_Distance(km)” which is the dependent variables which signifies the Driving Range predicito. We will do some Exploratory Data Analysis to understand the data and do some prediciton modeling.

#Read The Input Data

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.0.4
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
setwd("C:/Users/703026055/Desktop/Driving Range Prediction for Electric Vehicles")
df_ip = fread('Data-Final.csv')
```

Lets Check the Data Structure and checking for any null values

```
head(df_ip)
```

```
##    trip_distance(km) quantity(kWh)    tire_type city motor_way country_roads
## 1:                50         12.29 Winter tires    0         0             1
## 2:                43          8.68 Winter tires    0         1             1
## 3:                44          1.50 Winter tires    0         1             1
## 4:                76         14.44 Winter tires    0         1             0
## 5:                15          6.84 Winter tires    1         0             0
## 6:                63          7.10 Winter tires    0         1             0
##    driving_style consumption(kWh/100km) A/C park_heating avg_speed(km/h)
## 1:        Normal                   15.5   0            1              47
## 2:        Normal                   18.0   0            1              58
## 3:        Normal                   16.1   0            1              43
## 4:        Normal                   19.0   0            1              76
## 5:        Normal                   16.1   0            1              23
## 6:          Fast                   18.3   0            1              80
##    ecr_deviation
## 1:          -1.3
## 2:           1.2
## 3:          -0.7
## 4:           2.2
## 5:          -0.7
## 6:           1.5
```

```
dim(df_ip)
```

```
## [1] 3340   12
```

```
str(df_ip)
```

```
## Classes 'data.table' and 'data.frame':   3340 obs. of  12 variables:
##  $ trip_distance(km)     : num  50 43 44 76 15 63 85 71 76 80 ...
##  $ quantity(kWh)         : num  12.29 8.68 1.5 14.44 6.84 ...
##  $ tire_type             : chr  "Winter tires" "Winter tires" "Winter tires" "Winter tires"
...
##  $ city                  : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ motor_way             : int  0 1 1 1 0 1 1 0 0 0 ...
##  $ country_roads         : int  1 1 1 0 0 0 1 1 1 1 ...
##  $ driving_style         : chr  "Normal" "Normal" "Normal" "Normal" ...
##  $ consumption(kWh/100km): num  15.5 18 16.1 19 16.1 18.3 18 19.4 15.8 14 ...
##  $ A/C                   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ park_heating          : int  1 1 1 1 1 1 1 1 0 ...
##  $ avg_speed(km/h)       : num  47 58 43 76 23 80 47 45 78 52 ...
##  $ ecr_deviation         : num  -1.3 1.2 -0.7 2.2 -0.7 1.5 1.2 2.6 -1 -2.8 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
summary(df_ip)
```

```
##  trip_distance(km) quantity(kWh)     tire_type             city
##  Min.   :  0.50    Min.   : 0.010   Length:3340        Min.   :0.0000
##  1st Qu.: 20.00    1st Qu.: 1.790   Class :character   1st Qu.:0.0000
##  Median : 20.00    Median : 3.540   Mode  :character   Median :1.0000
##  Mean   : 42.08    Mean   : 5.911                      Mean   :0.6955
##  3rd Qu.: 57.00    3rd Qu.: 8.010                      3rd Qu.:1.0000
##  Max.   :642.00    Max.   :84.600                      Max.   :1.0000
##    motor_way      country_roads    driving_style      consumption(kWh/100km)
##  Min.   :0.0000   Min.   :0.0000   Length:3340        Min.   : 4.00
##  1st Qu.:0.0000   1st Qu.:0.0000   Class :character   1st Qu.:11.20
##  Median :1.0000   Median :1.0000   Mode  :character   Median :13.70
##  Mean   :0.6653   Mean   :0.5871                      Mean   :13.92
##  3rd Qu.:1.0000   3rd Qu.:1.0000                      3rd Qu.:16.80
##  Max.   :1.0000   Max.   :1.0000                      Max.   :76.00
##       A/C           park_heating    avg_speed(km/h)  ecr_deviation
##  Min.   :0.00000   Min.   :0.0000   Min.   :  2.00   Min.   :-12.80
##  1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.: 37.00   1st Qu.: -5.60
##  Median :0.00000   Median :0.0000   Median : 47.00   Median : -3.10
##  Mean   :0.04491   Mean   :0.1886   Mean   : 46.76   Mean   : -2.88
##  3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.: 56.00   3rd Qu.:  0.00
##  Max.   :1.00000   Max.   :1.0000   Max.   :100.00   Max.   : 59.20
```

```
#Checking for any Null Values.
#There are no Null values in the data
colSums(is.na(df_ip))
```

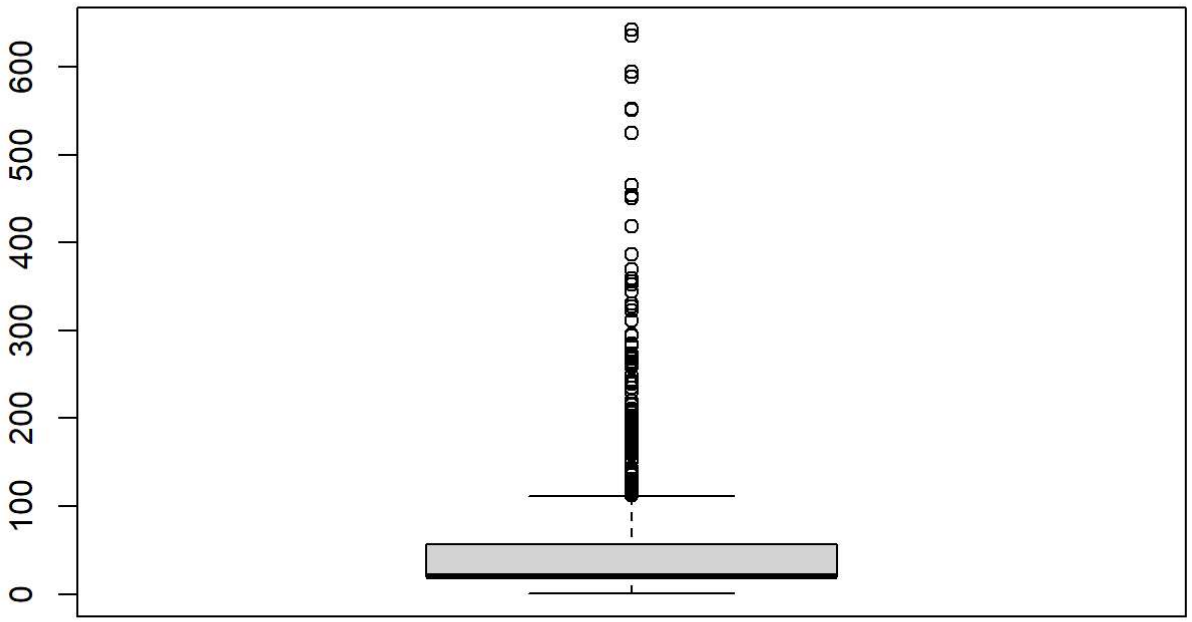
```
##      trip_distance(km)          quantity(kWh)              tire_type
##                      0                      0                      0
##                   city              motor_way         country_roads
##                      0                      0                      0
##          driving_style consumption(kWh/100km)                    A/C
##                      0                      0                      0
##           park_heating        avg_speed(km/h)          ecr_deviation
##                      0                      0                      0
```

#Data Conversion into Factors

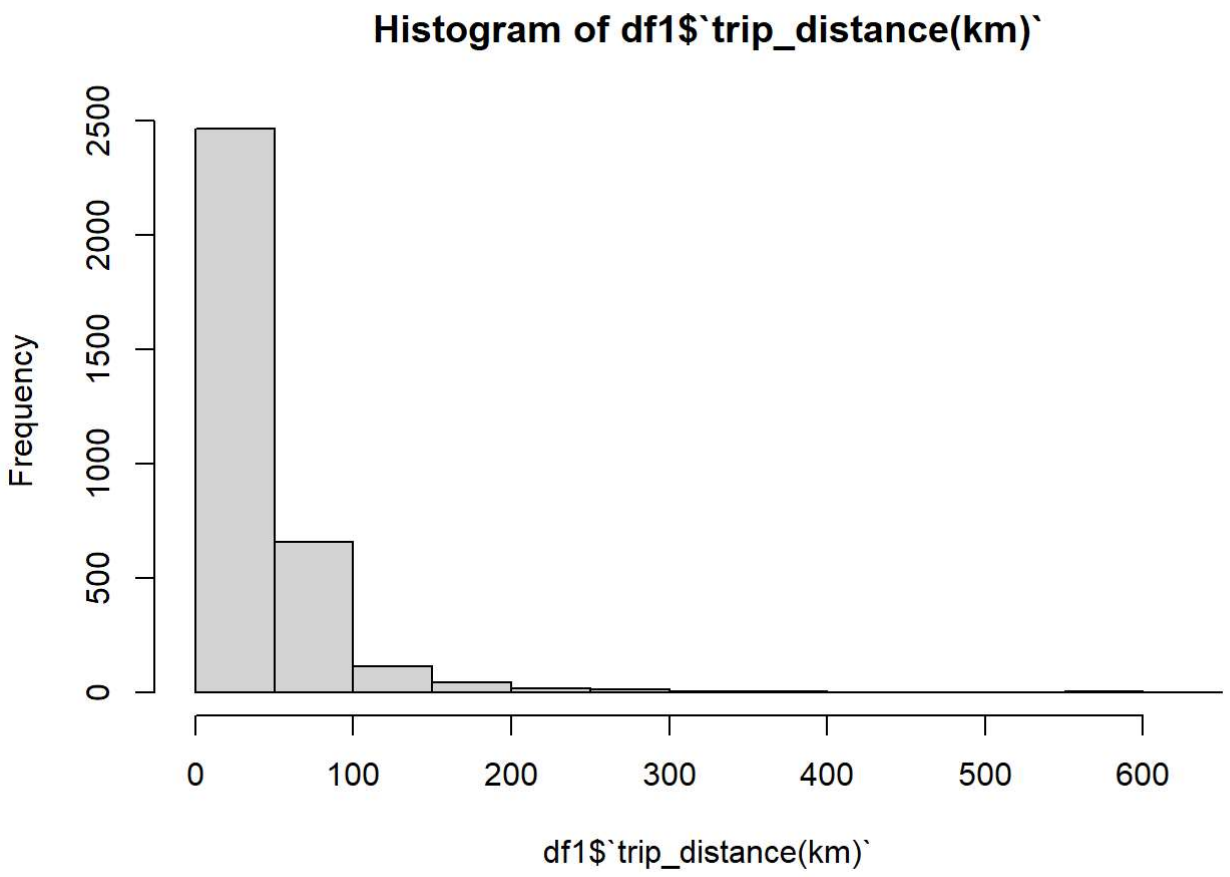
```
df_ip$city<-as.factor(df_ip$city)
df_ip$motor_way<-as.factor(df_ip$motor_way)
df_ip$country_roads<-as.factor(df_ip$country_roads)
df_ip$`A/C` <-as.factor(df_ip$`A/C`)
df_ip$park_heating<-as.factor(df_ip$park_heating)
df_ip<-as.data.frame(df_ip)
df1<-df_ip
```

Lets explore the Data using Univariate and Multivariate analysis

```
#DEPENDENT VARIABLE
boxplot(df1$`trip_distance(km)`)
```




```
hist(df1$`trip_distance(km)`)
```




```
#BOX and Histogram of the target variable show that there are lot of outliers which have to be r
emoved.  based on the plots, any value >150 can be removed.
df1<-df1[df1$`trip_distance(km)`<=100,]
#In the Quantiyt field, there seems to be some outliers.  Hence, for this model, anything less t
han 15 needs to be removed.
df1<-df1[df1$`quantity(kWh)` <=15,]

#UNI-VARIATE ANALYSIS
library(DescTools)
```

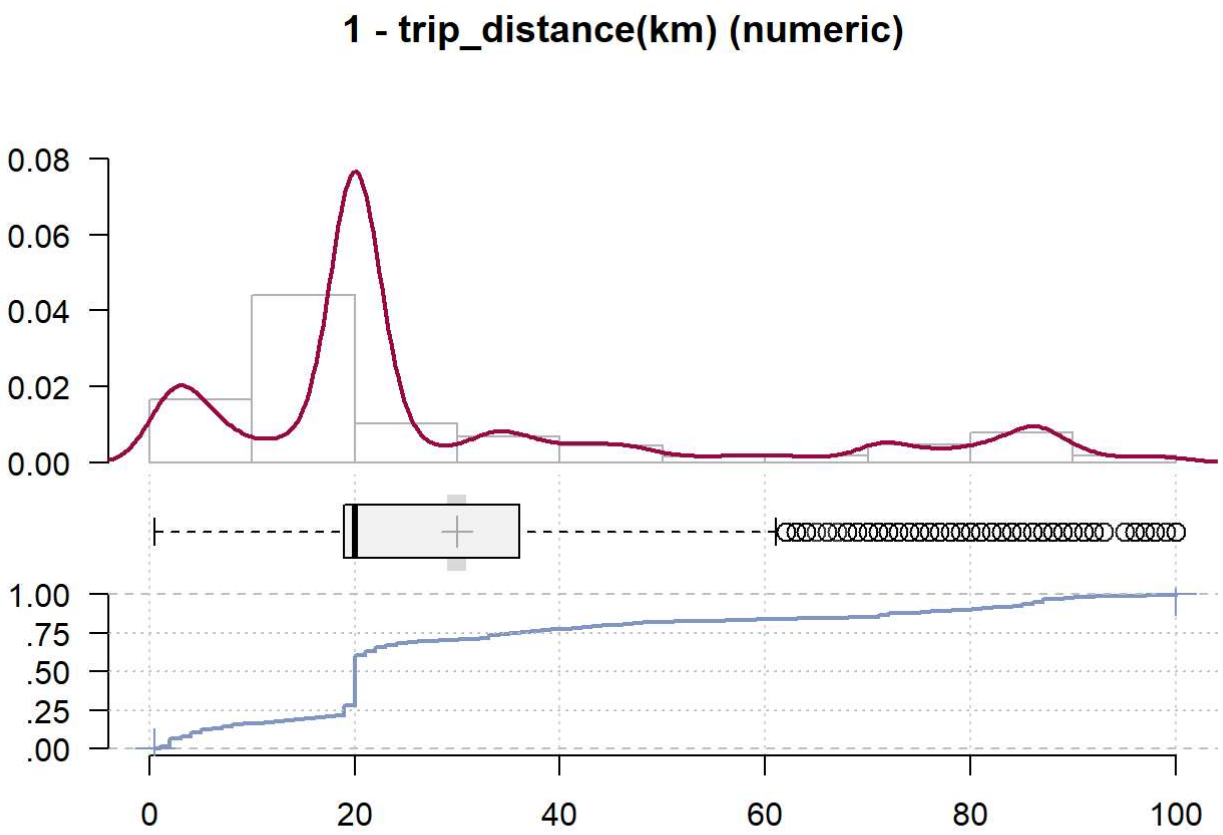
```
## Warning: package 'DescTools' was built under R version 4.0.4
```

```
##
## Attaching package: 'DescTools'
```

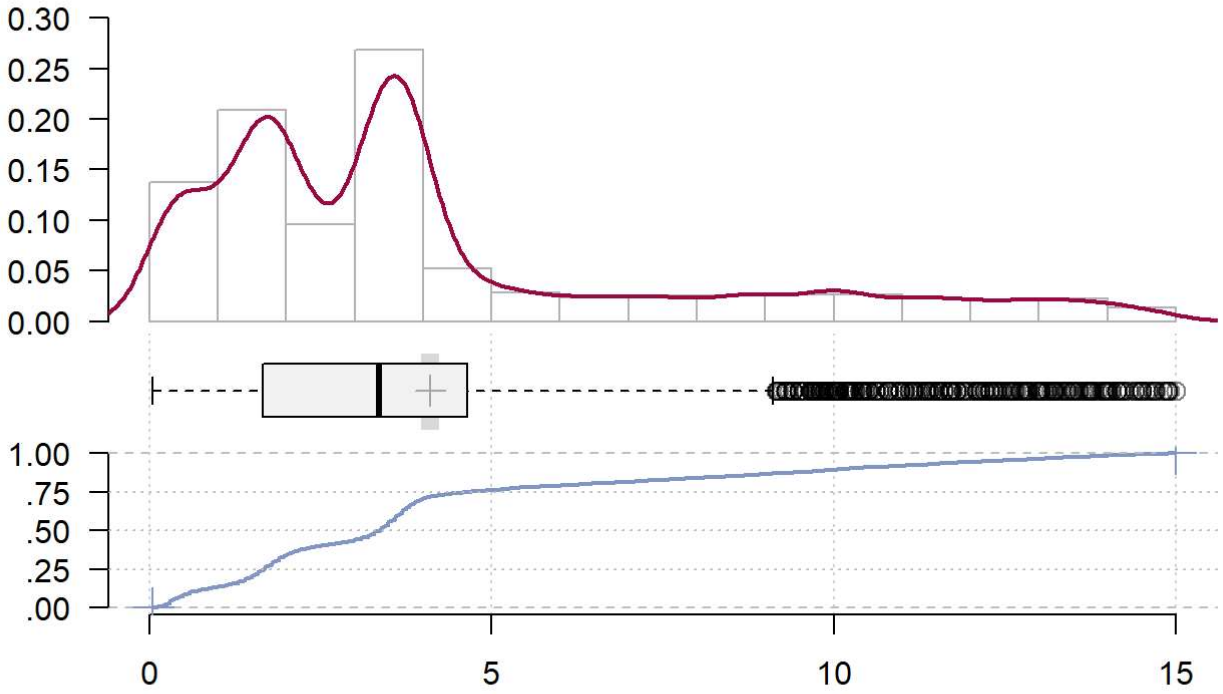
```
## The following object is masked from 'package:data.table':
##
##     %like%
```

```
Desc(df1)
```

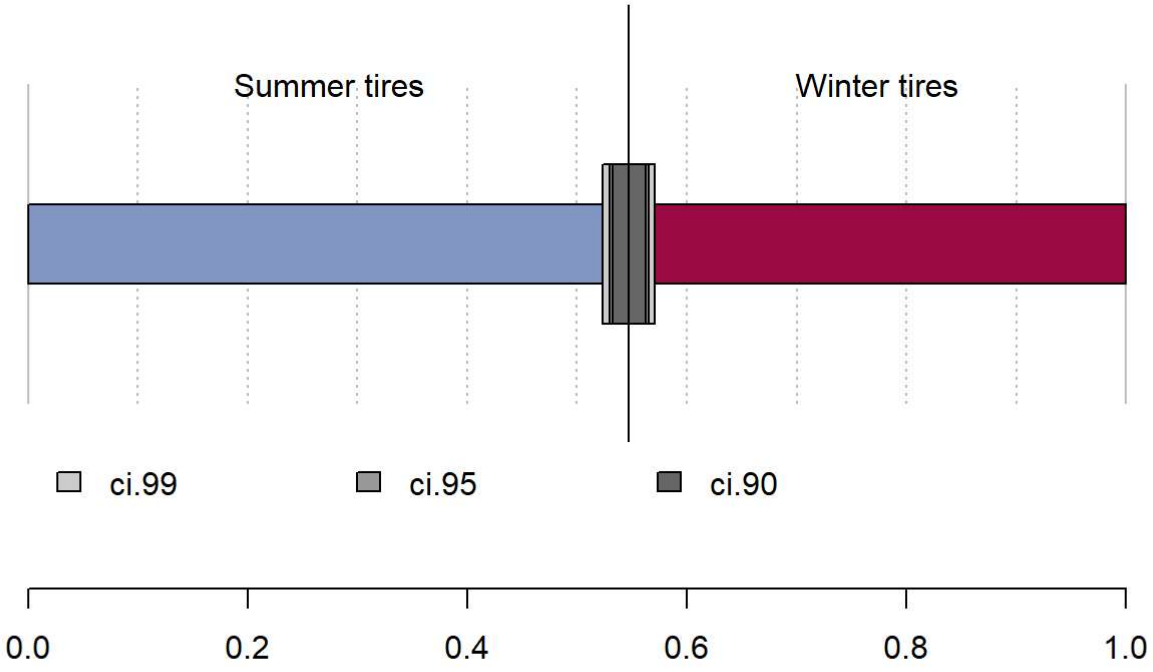
```
## ------------------------------------------------------------------------------
## Describe df1 (data.frame):
##
## data frame:  2953 obs. of  12 variables
##      2953 complete cases (100.0%)
##
##   Nr  ColName                 Class      NAs  Levels
##   1   trip_distance(km)       numeric    .
##   2   quantity(kWh)           numeric    .
##   3   tire_type               character  .
##   4   city                    factor     .    (2): 1-0, 2-1
##   5   motor_way               factor     .    (2): 1-0, 2-1
##   6   country_roads           factor     .    (2): 1-0, 2-1
##   7   driving_style           character  .
##   8   consumption(kWh/100km)  numeric    .
##   9   A/C                     factor     .    (2): 1-0, 2-1
##   10  park_heating            factor     .    (2): 1-0, 2-1
##   11  avg_speed(km/h)         numeric    .
##   12  ecr_deviation           numeric    .
##
##
## ------------------------------------------------------------------------------
## 1 - trip_distance(km) (numeric)
##
##   length       n    NAs  unique     0s   mean  meanCI'
##    2'953   2'953      0     100      0  29.95   29.05
##          100.0%   0.0%           0.0%          30.86
##
##      .05     .10    .25  median    .75    .90     .95
##     2.00    4.00  19.00   20.00  36.00  80.00   86.00
##
##    range      sd  vcoef     mad    IQR   skew    kurt
##    99.50   25.16   0.84    7.41  17.00   1.30    0.55
##
## lowest : 0.5, 1.0 (50), 2.0 (145), 3.0 (51), 4.0 (77)
## highest: 96.0 (6), 97.0 (6), 98.0 (6), 99.0 (3), 100.0 (9)
##
## heap(?): remarkable frequency (32.8%) for the mode(s) (= 20)
##
## ' 95%-CI (classic)
```



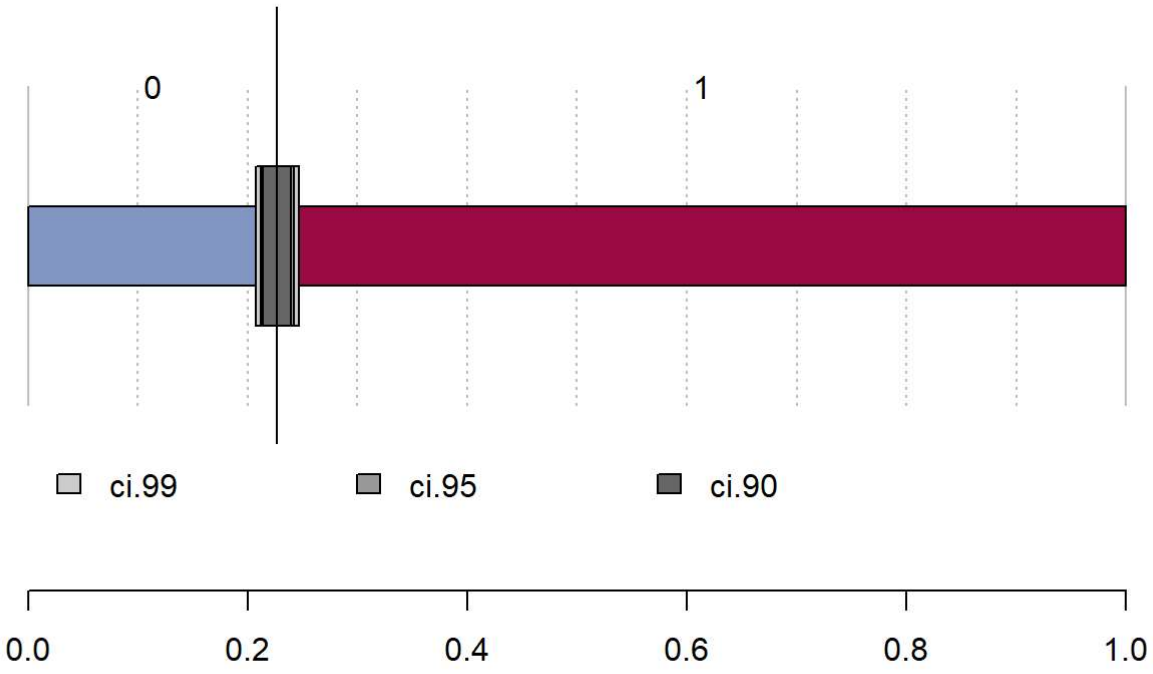

```
## ------------------------------------------------------------------------------
## 2 - quantity(kWh) (numeric)
##
##   length       n    NAs  unique    0s   mean  meanCI'
##    2'953   2'953      0     805     0   4.10    3.98
##           100.0%   0.0%          0.0%           4.23
##
##      .05     .10    .25  median   .75    .90     .95
##     0.33    0.60   1.66    3.36  4.65  10.18   12.28
##
##    range      sd  vcoef     mad   IQR   skew    kurt
##    14.95    3.54   0.86    2.46  2.99   1.34    0.95
##
## lowest : 0.05, 0.09 (3), 0.1 (15), 0.11 (6), 0.12 (3)
## highest: 14.81 (2), 14.85, 14.88 (4), 14.91, 15.0
##
## ' 95%-CI (classic)
```



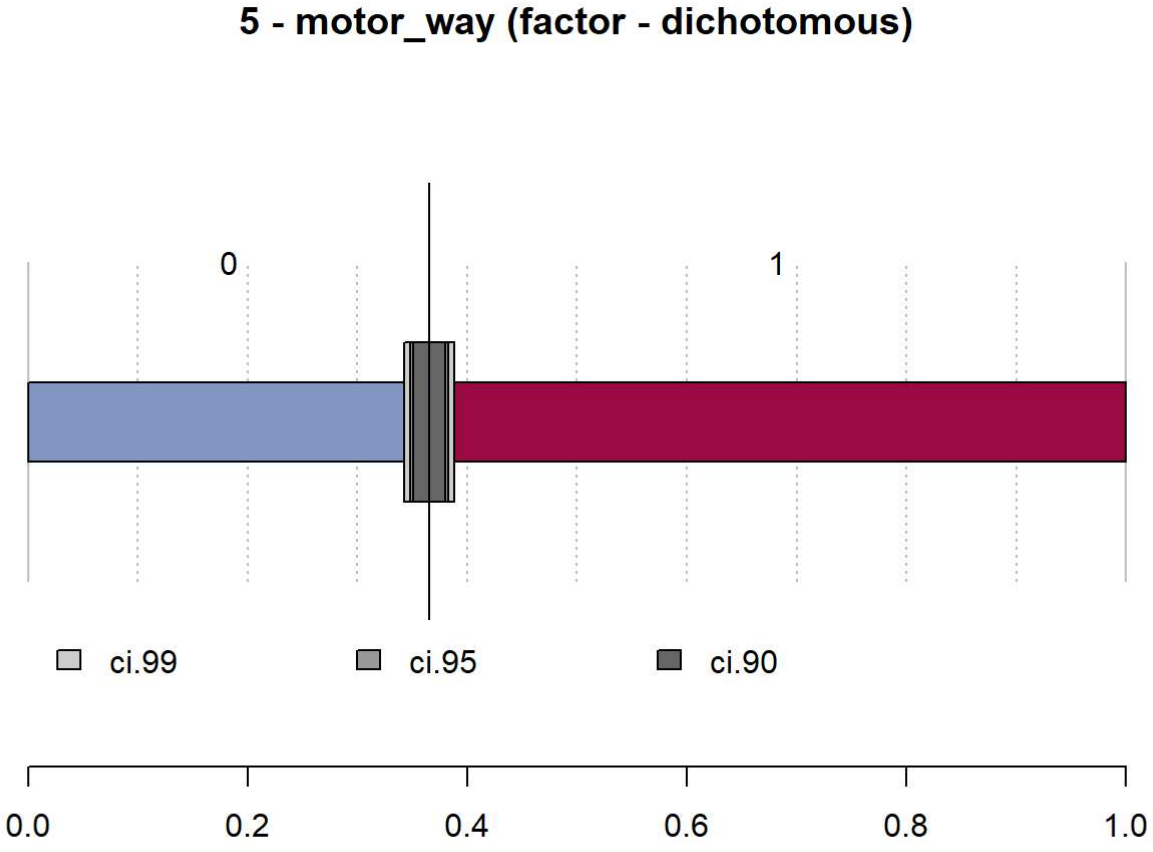

```
## ----------------------------------------------------------------------------
## 3 - tire_type (character - dichotomous)
##
##   length      n    NAs unique
##    2'953  2'953      0      2
##          100.0%   0.0%
##
##                freq   perc  lci.95  uci.95'
## Summer tires  1'616  54.7%   52.9%   56.5%
## Winter tires  1'337  45.3%   43.5%   47.1%
##
## ' 95%-CI (Wilson)
```



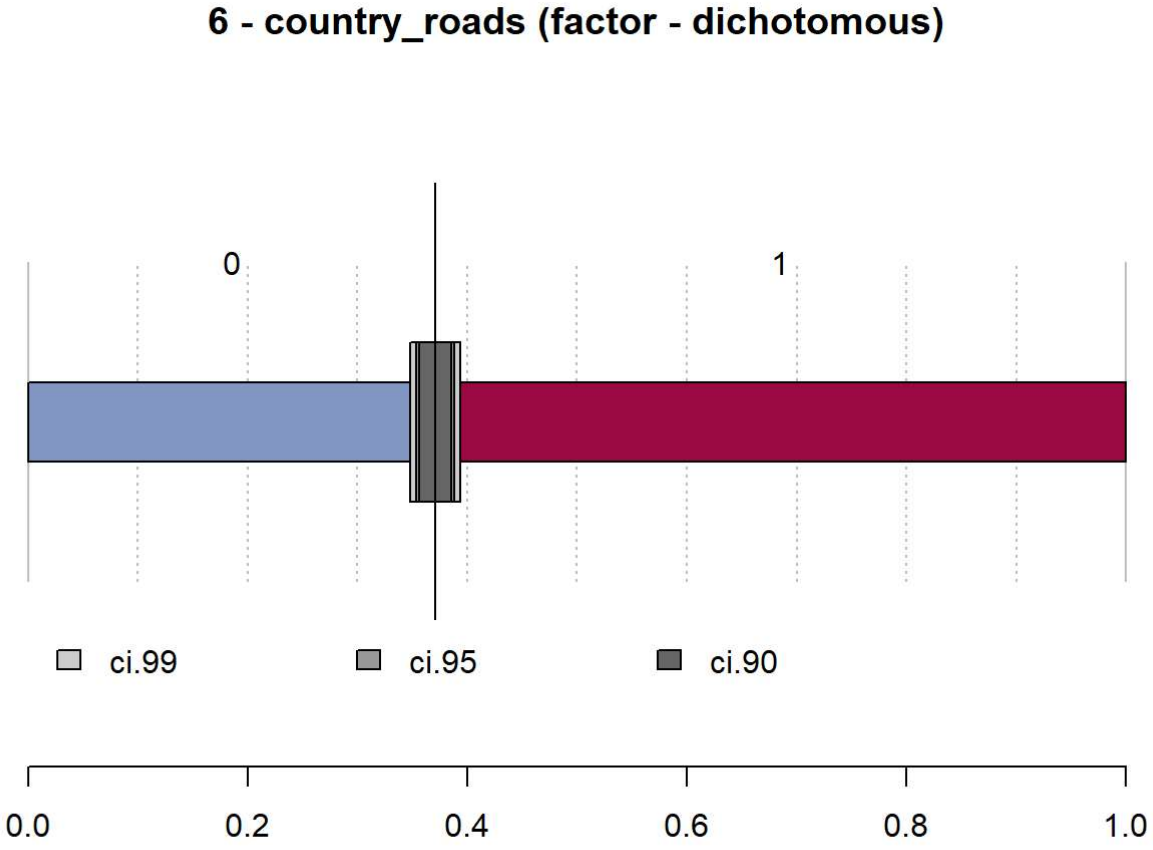

```
## ------------------------------------------------------------------------------
## 4 - city (factor - dichotomous)
##
##   length      n    NAs unique
##    2'953  2'953      0      2
##          100.0%   0.0%
##
##     freq   perc  lci.95  uci.95'
## 0    669  22.7%   21.2%   24.2%
## 1  2'284  77.3%   75.8%   78.8%
##
## ' 95%-CI (Wilson)
```



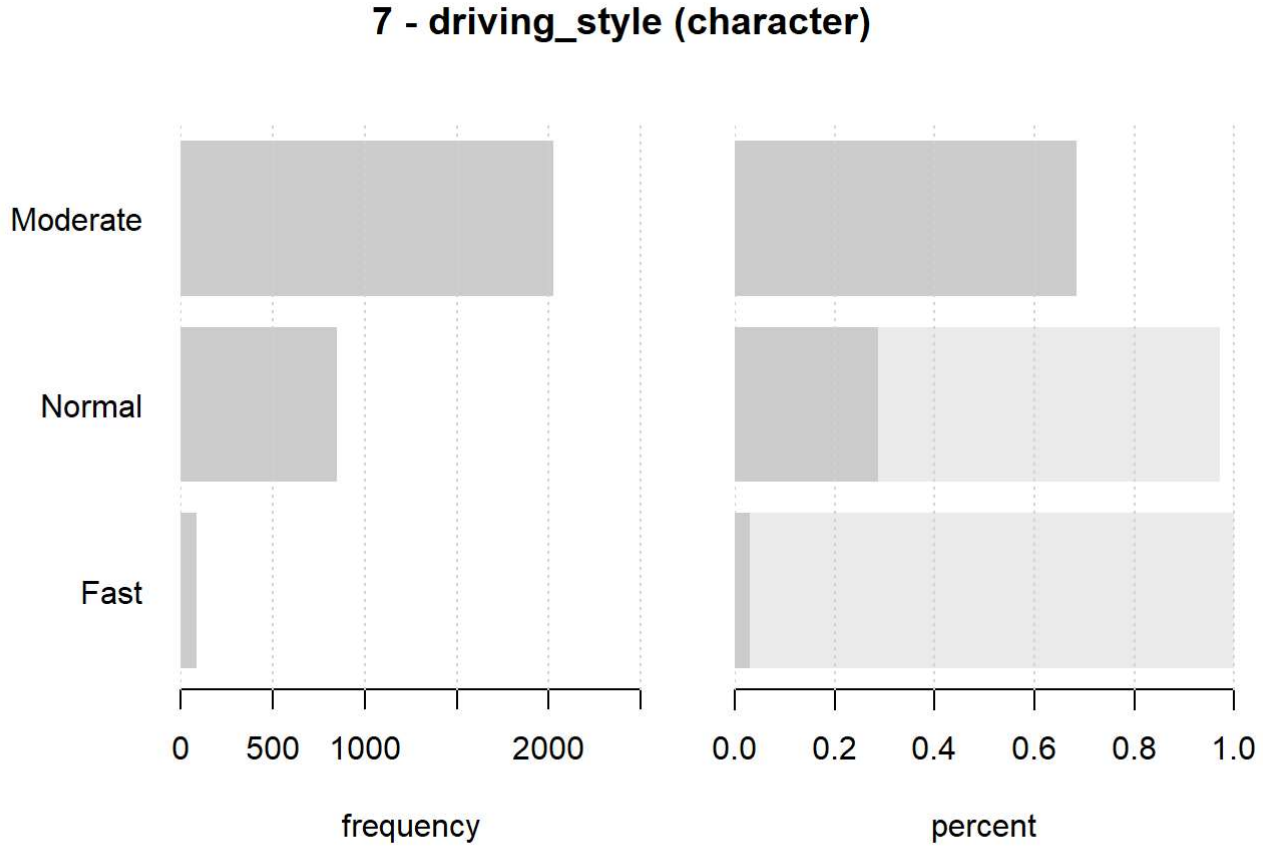

```
## ------------------------------------------------------------------------------
## 5 - motor_way (factor - dichotomous)
##
##   length      n    NAs unique
##    2'953  2'953      0      2
##          100.0%   0.0%
##
##     freq   perc  lci.95  uci.95'
## 0  1'078  36.5%   34.8%   38.3%
## 1  1'875  63.5%   61.7%   65.2%
##
## ' 95%-CI (Wilson)
```



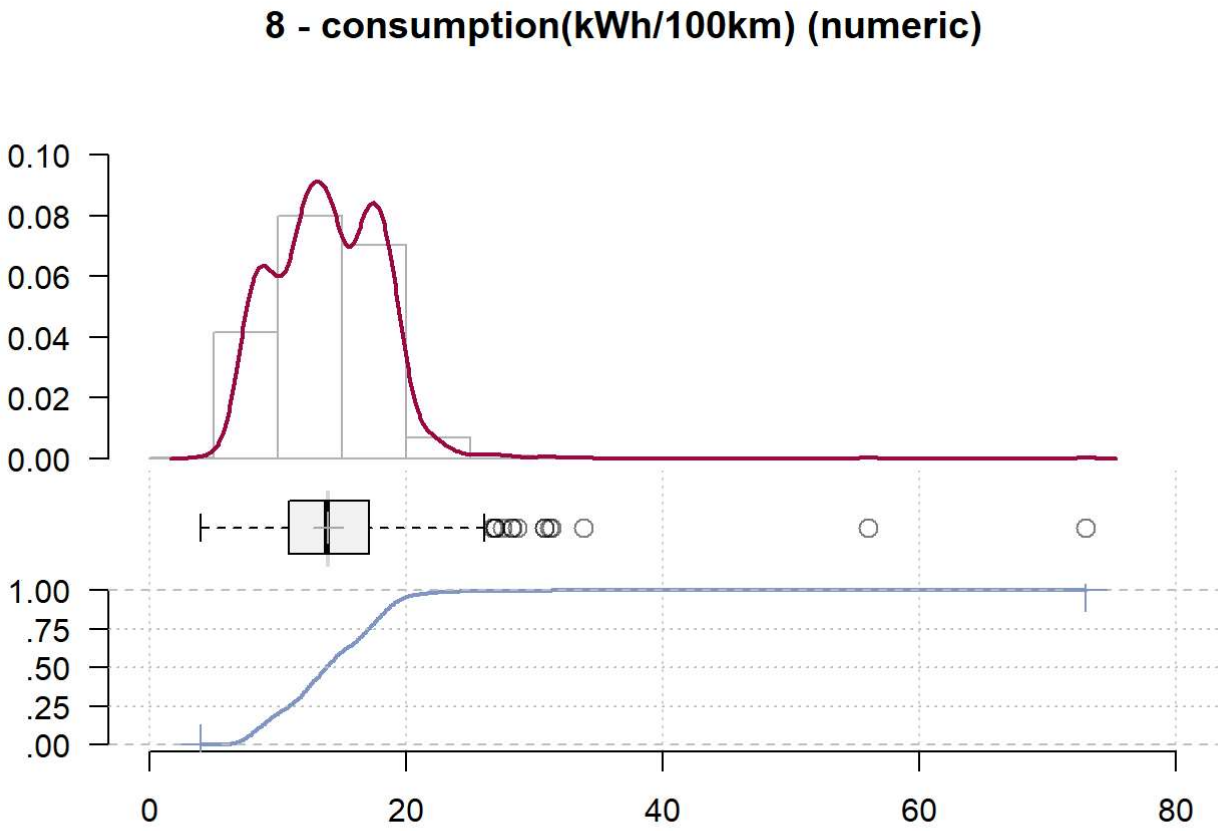

```
## ------------------------------------------------------------------------------
## 6 - country_roads (factor - dichotomous)
##
##   length      n    NAs unique
##    2'953  2'953      0      2
##          100.0%   0.0%
##
##     freq   perc  lci.95  uci.95'
## 0  1'094  37.0%   35.3%   38.8%
## 1  1'859  63.0%   61.2%   64.7%
##
## ' 95%-CI (Wilson)
```



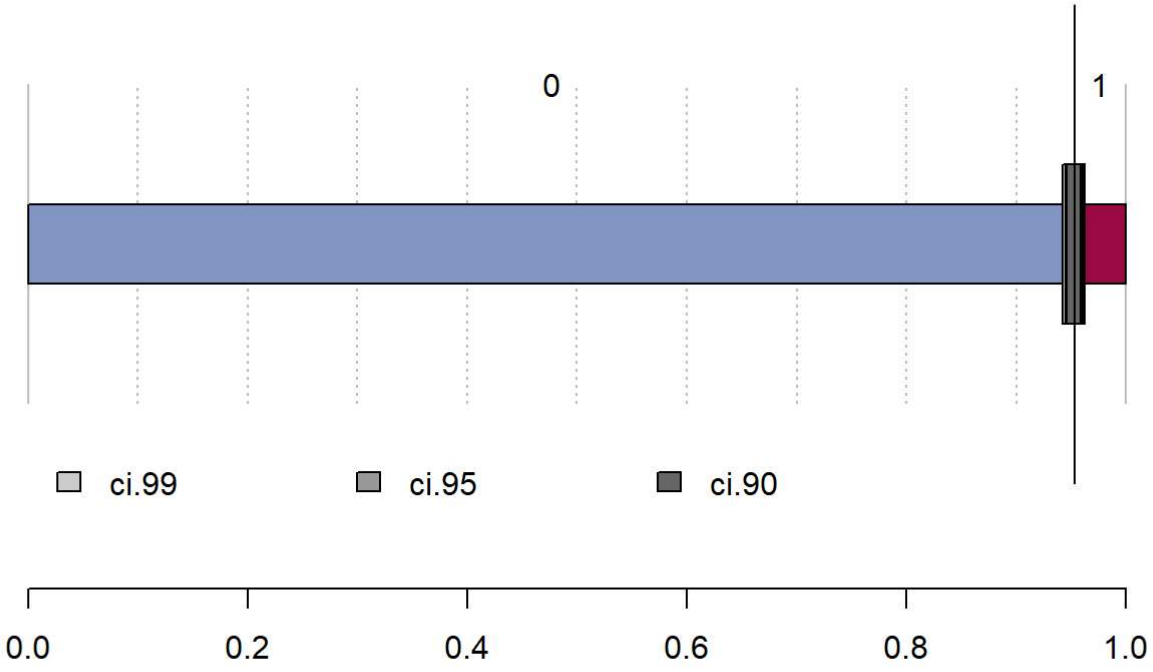

```
## ------------------------------------------------------------------------------
## 7 - driving_style (character)
##
##   length      n    NAs unique levels  dupes
##    2'953  2'953      0      3      3      y
##          100.0%   0.0%
##
##       level   freq   perc  cumfreq  cumperc
## 1  Moderate  2'022  68.5%    2'022    68.5%
## 2    Normal    847  28.7%    2'869    97.2%
## 3      Fast     84   2.8%    2'953   100.0%
```



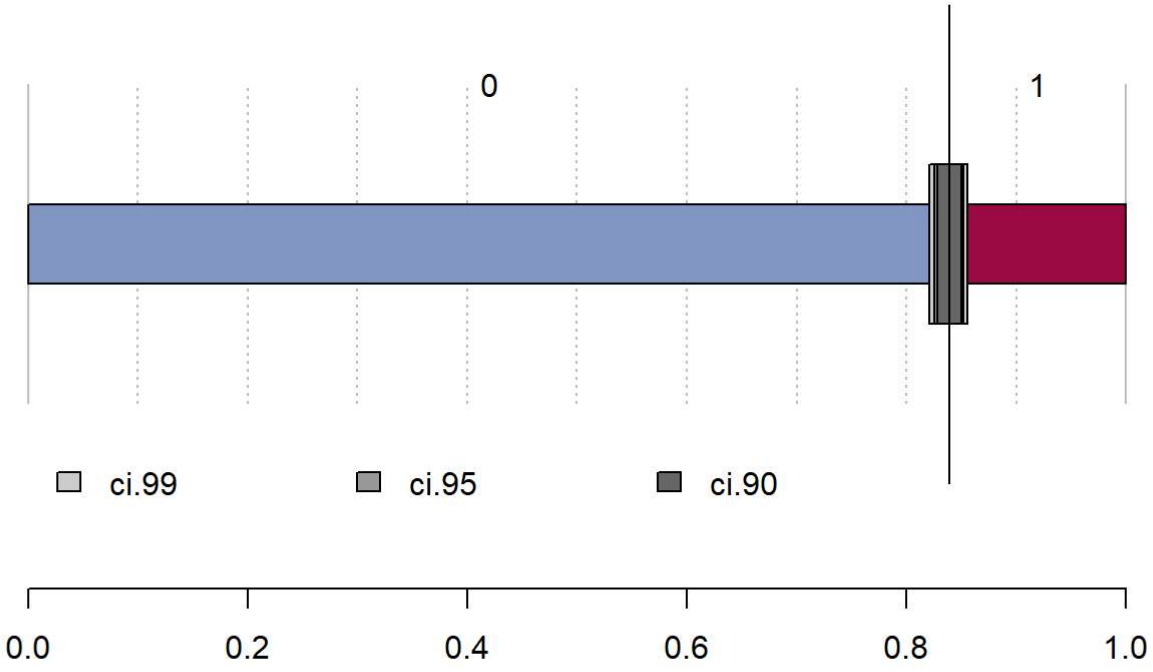

```
## ------------------------------------------------------------------------------
## 8 - consumption(kWh/100km) (numeric)
##
##   length       n    NAs  unique     0s   mean  meanCI'
##    2'953   2'953      0     213      0  13.91   13.76
##           100.0%   0.0%           0.0%          14.07
##
##      .05     .10    .25  median    .75    .90     .95
##     7.60    8.32  10.90   13.80  17.10  18.80   19.74
##
##    range      sd  vcoef     mad    IQR   skew    kurt
##    69.00    4.24   0.30    4.60   6.20   1.43   15.50
##
## lowest : 4.0, 4.5 (2), 5.2, 5.4, 5.5 (2)
## highest: 31.2, 31.3, 33.8, 56.0, 73.0
##
## ' 95%-CI (classic)
```



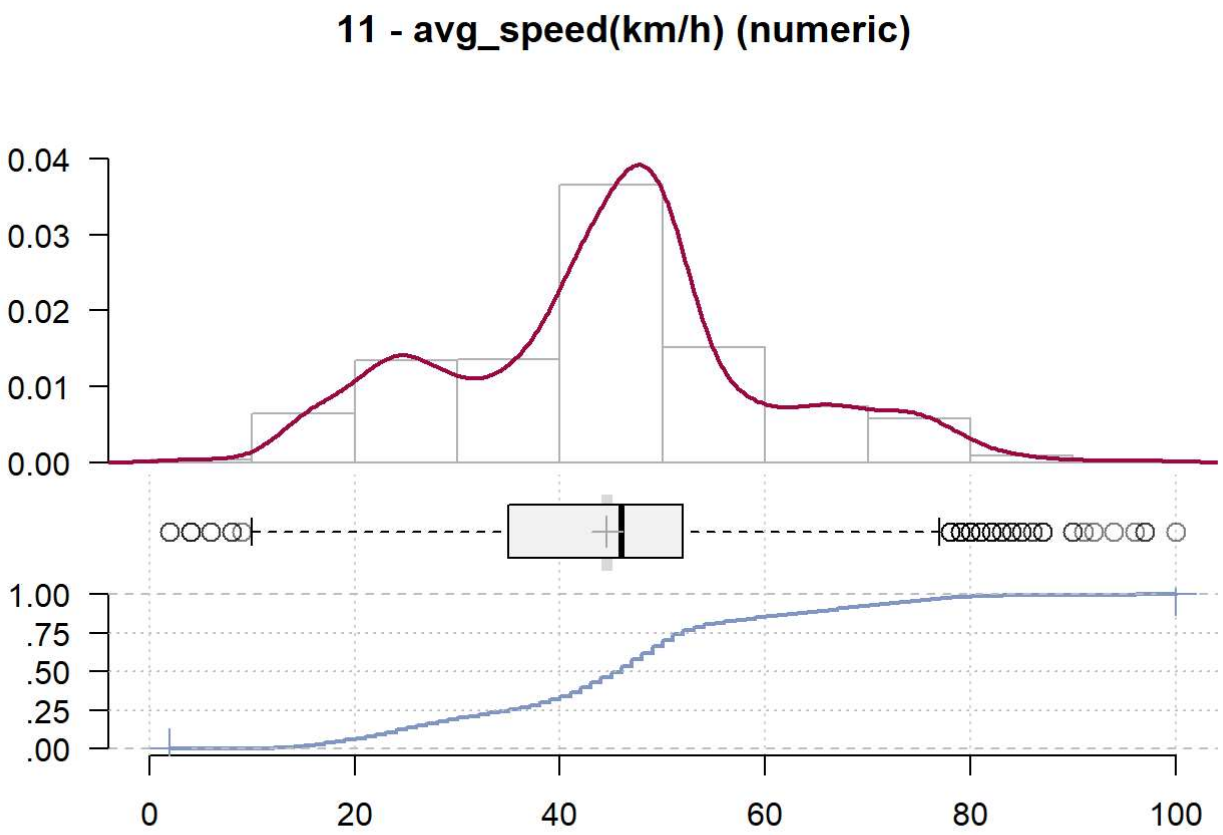

```
## ------------------------------------------------------------------------------
## 9 - A/C (factor - dichotomous)
##
##   length      n    NAs unique
##    2'953  2'953      0      2
##          100.0%   0.0%
##
##     freq   perc  lci.95  uci.95'
## 0  2'815  95.3%   94.5%   96.0%
## 1    138   4.7%    4.0%    5.5%
##
## ' 95%-CI (Wilson)
```



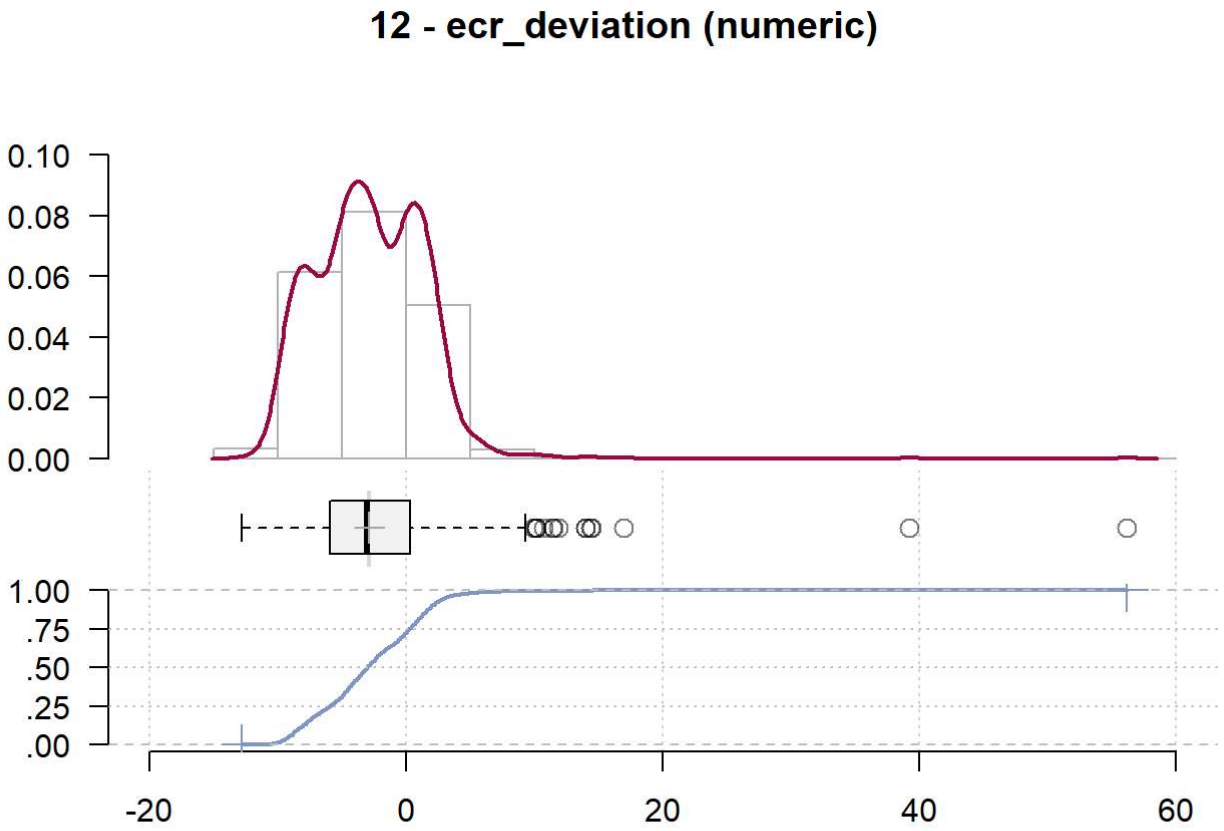

```
## ------------------------------------------------------------------------------
## 10 - park_heating (factor - dichotomous)
##
##   length      n    NAs unique
##    2'953  2'953      0      2
##          100.0%   0.0%
##
##     freq   perc  lci.95  uci.95'
## 0  2'479  83.9%   82.6%   85.2%
## 1    474  16.1%   14.8%   17.4%
##
## ' 95%-CI (Wilson)
```




```
## ------------------------------------------------------------------------------
## 11 - avg_speed(km/h) (numeric)
##
##   length       n    NAs  unique     0s   mean  meanCI'
##    2'953   2'953      0      99      0  44.57   44.01
##           100.0%   0.0%           0.0%          45.13
##
##      .05     .10    .25  median    .75    .90     .95
##    19.00   23.00  35.00   46.00  52.00  66.00   73.00
##
##    range      sd  vcoef     mad    IQR   skew    kurt
##    98.00   15.57   0.35   11.86  17.00   0.17    0.04
##
## lowest : 2.0 (2), 4.0 (3), 6.0 (2), 8.0 (2), 9.0
## highest: 92.0, 94.0, 96.0, 97.0 (2), 100.0
##
## ' 95%-CI (classic)
```




```
## ------------------------------------------------------------------------------
## 12 - ecr_deviation (numeric)
##
##   length       n    NAs  unique    0s   mean  meanCI'
##    2'953   2'953      0     213    19  -2.89   -3.04
##           100.0%   0.0%          0.6%          -2.73
##
##      .05     .10    .25  median   .75    .90     .95
##    -9.20   -8.48  -5.90   -3.00  0.30   2.00    2.94
##
##    range      sd  vcoef     mad   IQR   skew    kurt
##    69.00    4.24  -1.47    4.60  6.20   1.43   15.50
##
## lowest : -12.80, -12.30 (2), -11.60, -11.40, -11.30 (2)
## highest: 14.4, 14.5, 17.0, 39.2, 56.2
##
## ' 95%-CI (classic)
```




```
#BI-VARIATE ANALYSIS
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.4
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
library(DT)
```

```
## Warning: package 'DT' was built under R version 4.0.4
```

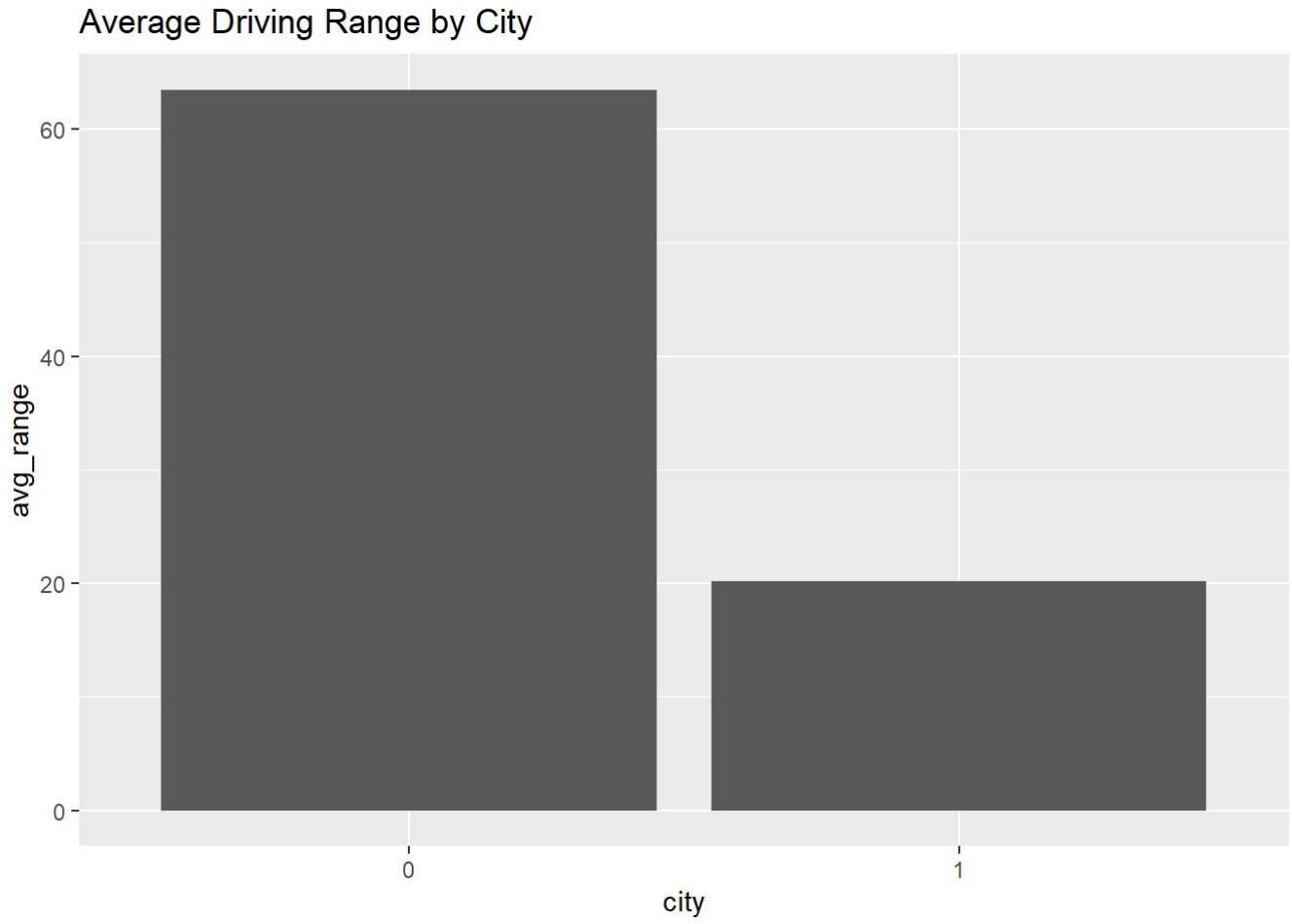
```
#Average Driving Range by City
#From the summaries below we can see that City has high predictive power.  However we also have to check for classification imbalance
Range_City= df1 %>%
  group_by(city) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_City)
```

Show 10 entries Search:

| | city | avg_range |
|---|---|---|
| 1 | 0 | 63.4723467862481 |
| 2 | 1 | 20.1355078809107 |

Showing 1 to 2 of 2 entries Previous 1 Next

```
plt_city = df1 %>% group_by(city) %>% summarise(avg_range = mean(`trip_distance(km)`)) %>%
  ggplot(aes(x=city,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by City")
plot(plt_city)
```



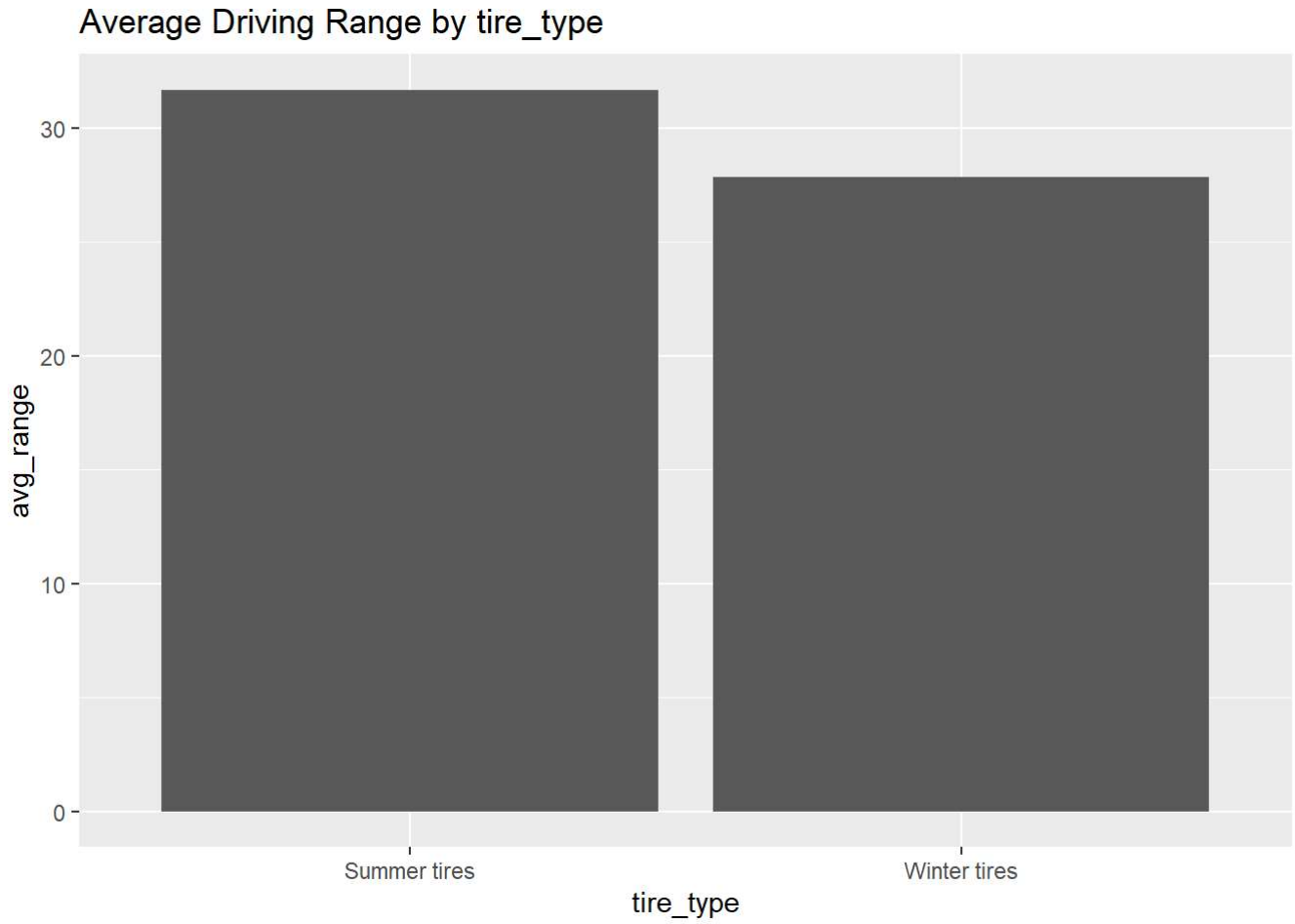

```
#Average Driving Range by Tire Type
#As it can be seen, there is no much difference in the average driving Range with the Tire_Type.
This variable can be ignored in the model
Range_tiretype= df1 %>%
  group_by(tire_type) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_tiretype)
```

**Show** 10 **entries** **Search:**

| | tire_type | avg_range |
|---|---|---|
| 1 | Summer tires | 31.6939975247525 |
| 2 | Winter tires | 27.8496634255797 |

Showing 1 to 2 of 2 entries Previous 1 Next

```
plt_tire = df1 %>% group_by(tire_type) %>% summarise(avg_range = mean(`trip_distance(km)`)) %>%
  ggplot(aes(x=tire_type,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by tire_type")
plot(plt_tire)
```



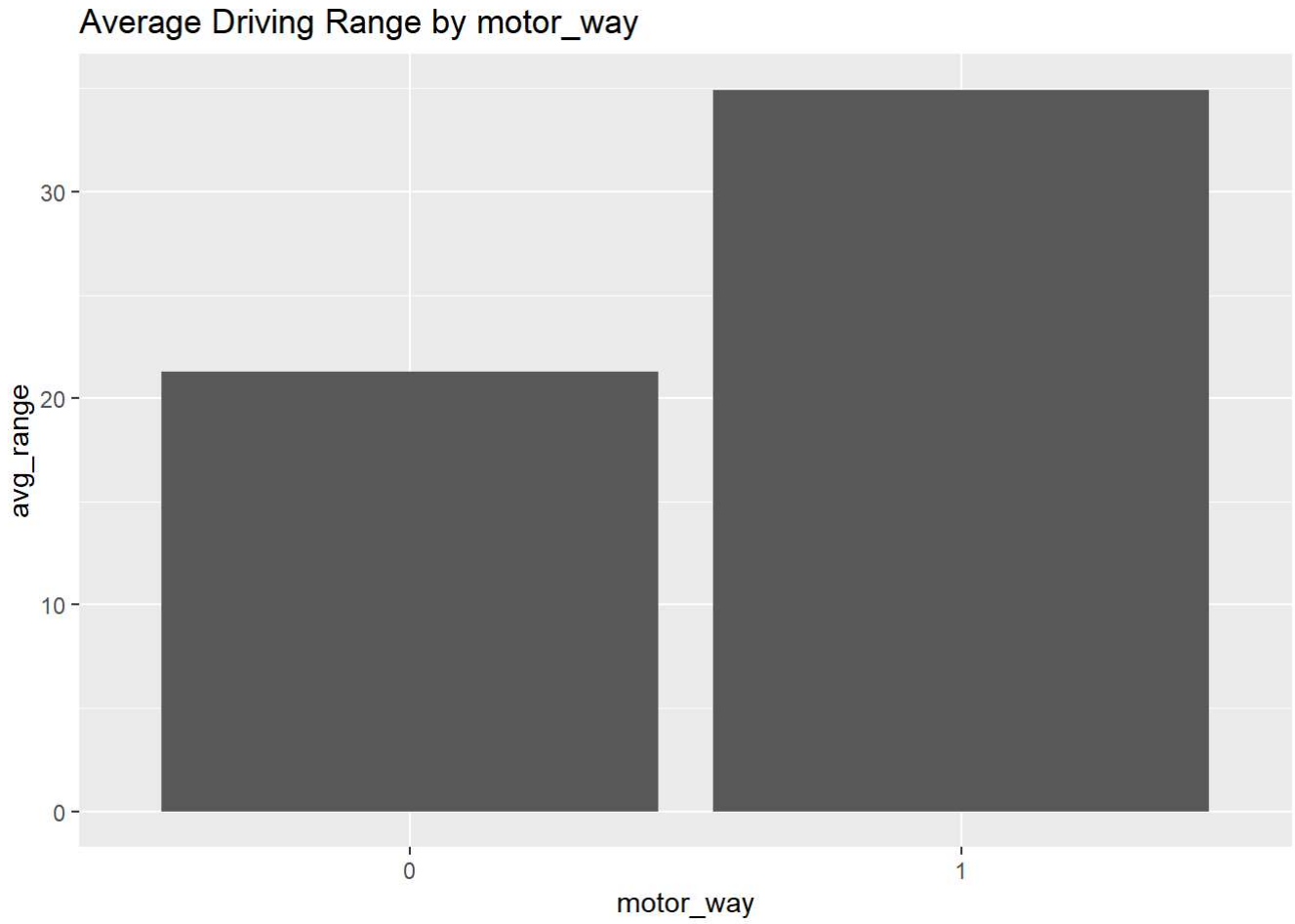

```
#Average Driving Range by Motor Way
#Motor way has a good predictive power for Driving Rage.  This variable will be used for the mod
el
Range_MotorWay= df1 %>%
  group_by(motor_way) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_MotorWay)
```

Show 10 entries Search:

| | motor_way | avg_range |
|---|---|---|
| 1 | 1 | 34.944 |
| 2 | 0 | 21.2731910946197 |

Showing 1 to 2 of 2 entries Previous 1 Next

```
plt_motor = df1 %>% group_by(motor_way) %>% summarise(avg_range = mean(`trip_distance(km)`)) %>%
  ggplot(aes(x=motor_way,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by motor_way")
plot(plt_motor)
```



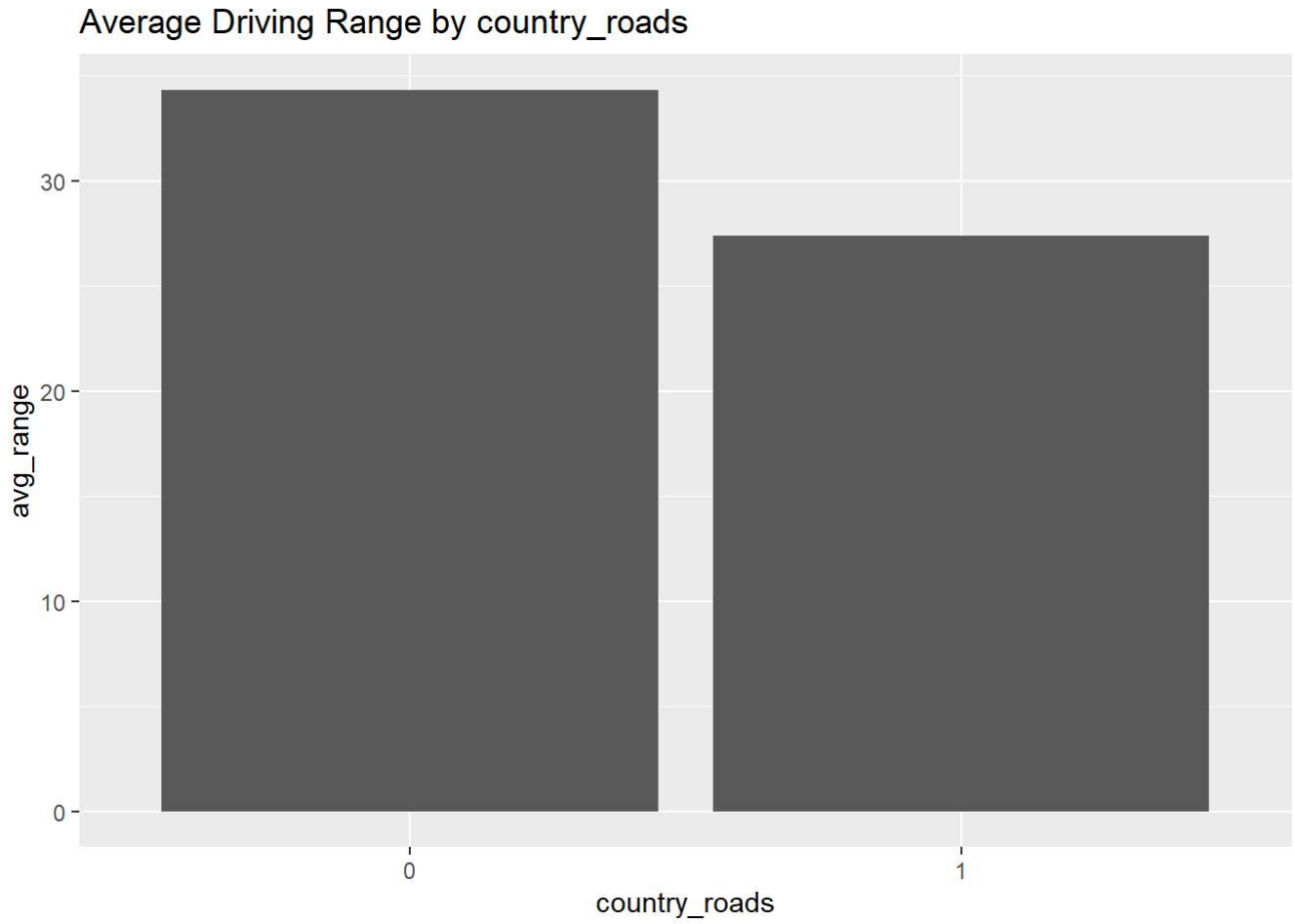

```
#Average Driving Range by Country Road
#From the Summaries below, Country Road will be used in the model as it has significant results.
Range_CountryRoad= df1 %>%
  group_by(country_roads) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_CountryRoad)
```

Show 10 entries Search:

| | country_roads | avg_range |
|---|---|---|
| 1 | 0 | 34.3350091407678 |
| 2 | 1 | 27.3749327595481 |

Showing 1 to 2 of 2 entries Previous 1 Next

```
plt_CountryRoad = df1 %>% group_by(country_roads) %>% summarise(avg_range = mean(`trip_distance
(km)`)) %>%
  ggplot(aes(x=country_roads,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by country_roads")
plot(plt_CountryRoad)
```




```
#Average Driving Range by Driving Style
#as it can be seen, the average driving range reduces as the driving style moves from Fast, Mode
rate to Normal.  Hence, this variable should be used for the model.
Range_DrivingSytle= df1 %>%
  group_by(driving_style) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_DrivingSytle)
```

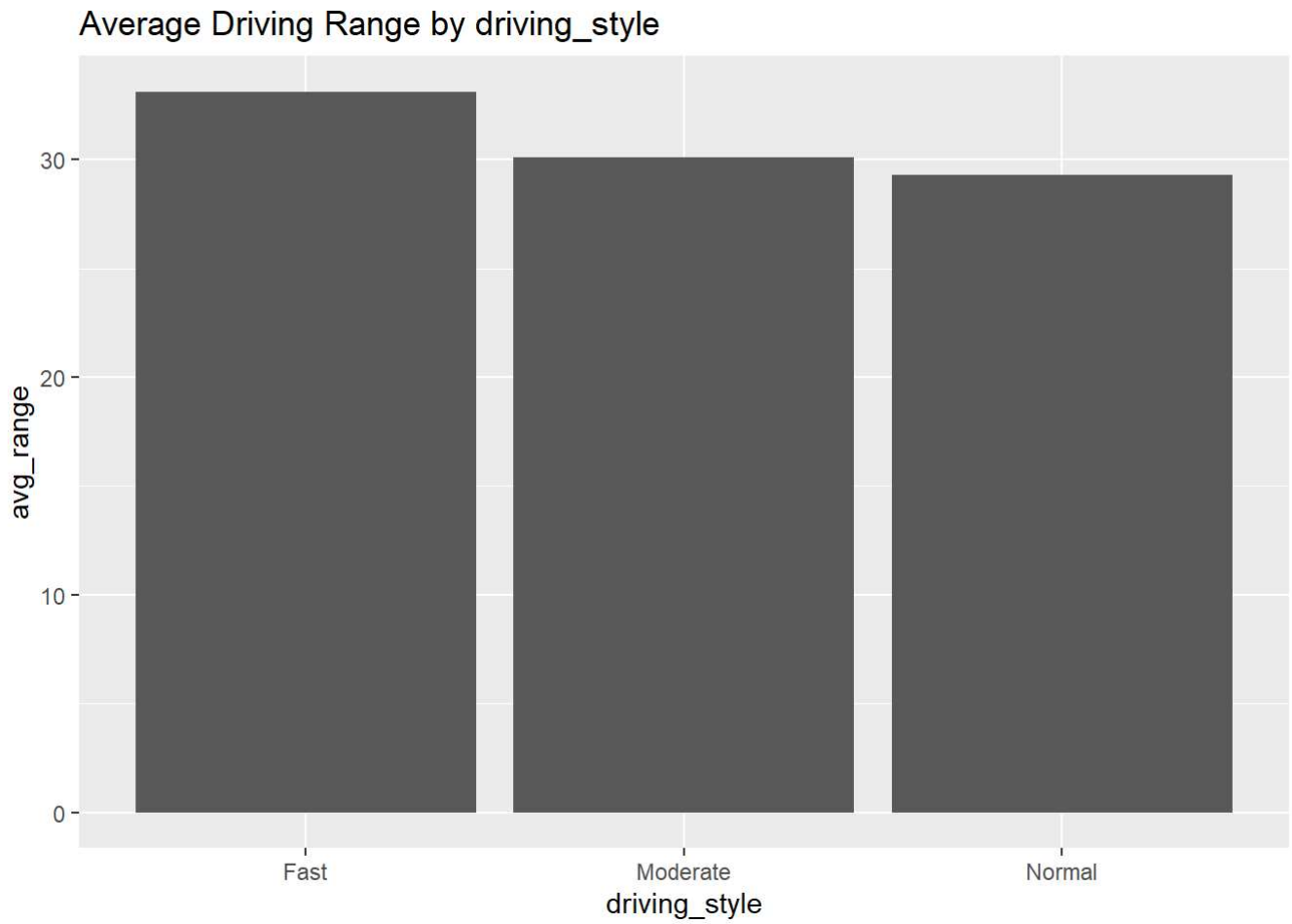
**Show** 10 **entries** **Search:**

| | driving_style | avg_range |
|---|---|---|
| 1 | Fast | 33.1190476190476 |
| 2 | Moderate | 30.1043521266073 |

| | driving_style | avg_range |
|---|---|---|
| 3 | Normal | 29.2792207792208 |

Showing 1 to 3 of 3 entries Previous 1 Next

```
plt_dvngstyle = df1 %>% group_by(driving_style) %>% summarise(avg_range = mean(`trip_distance(km)`)) %>%
  ggplot(aes(x=driving_style,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by driving_style")
plot(plt_dvngstyle)
```



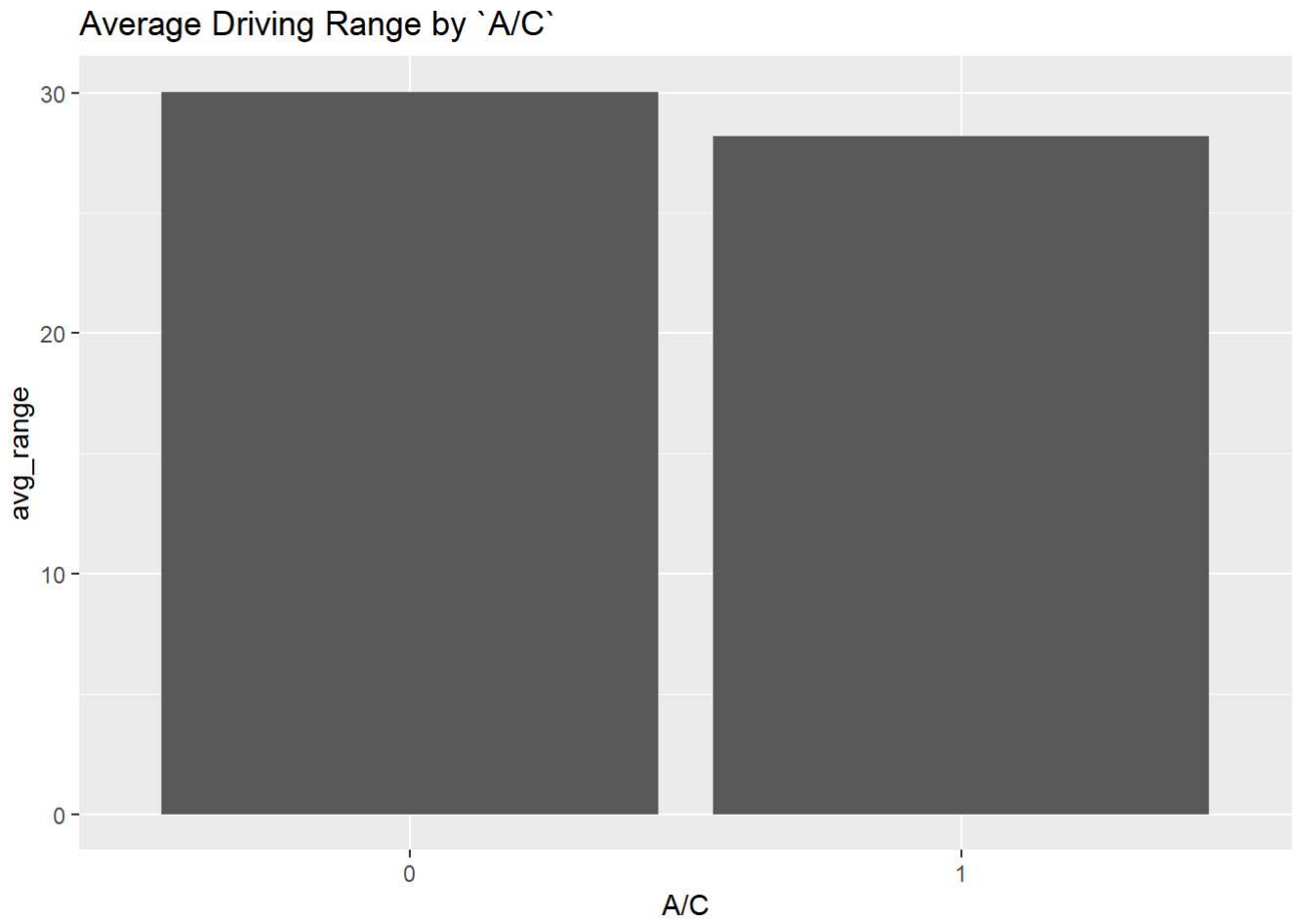

```
#Average Driving Range by A/C
#As seen below, A/C is an important variable.  However, as seen above from its univariate analysis, this has a hugh classification imbalance issue.  Hence this variable will NOT be considered.
Range_AC= df1 %>%
  group_by(`A/C`) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_AC)
```

Show 10 entries Search:

| | A/C | avg_range |
|---|---|---|
| 1 | 0 | 30.0399644760213 |
| 2 | 1 | 28.1884057971014 |

Showing 1 to 2 of 2 entries Previous 1 Next

```
plt_AC = df1 %>% group_by(`A/C`) %>% summarise(avg_range = mean(`trip_distance(km)`)) %>%
  ggplot(aes(x=`A/C`,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by `A/C`")
plot(plt_AC)
```



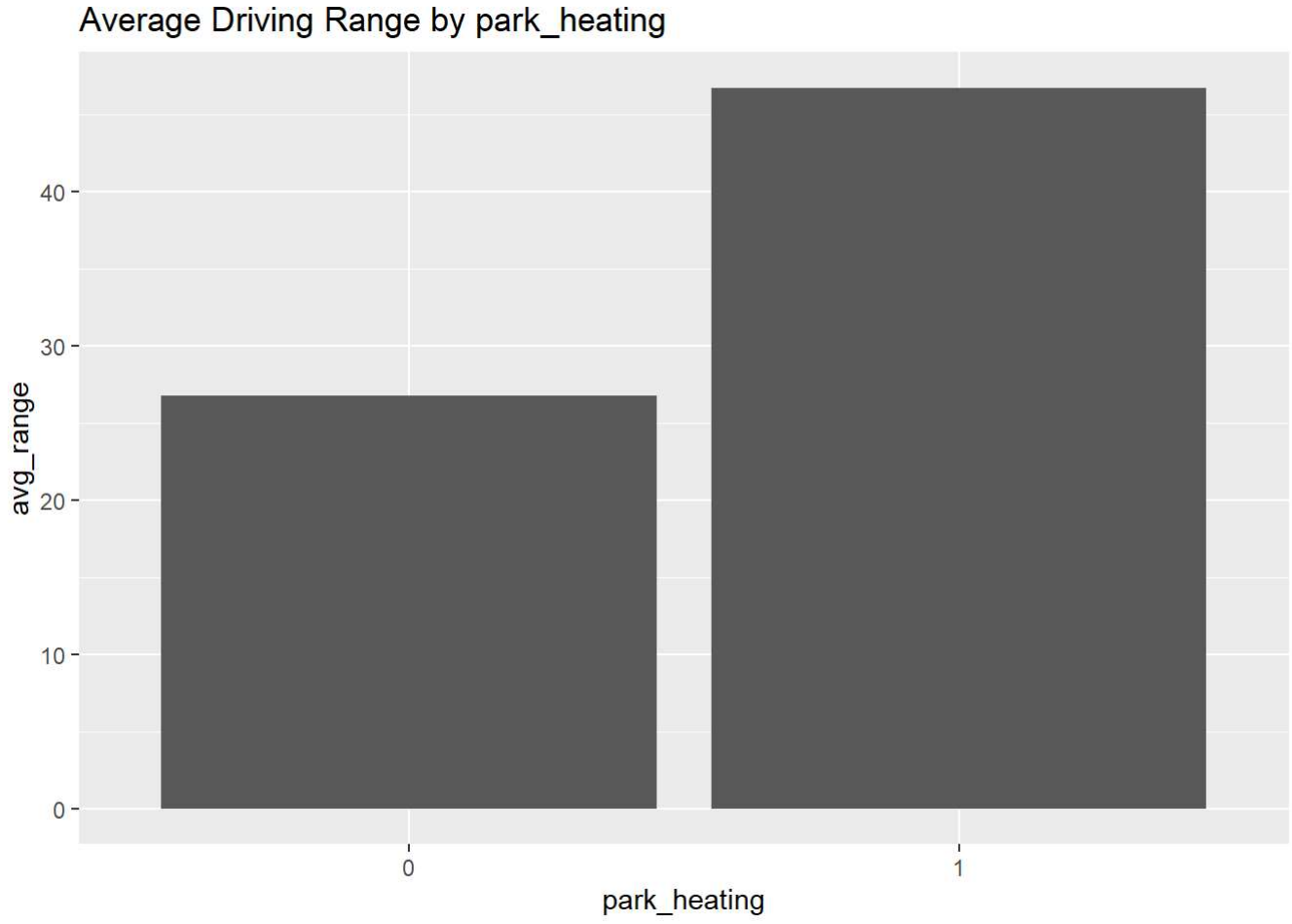

```
#Average Driving Range by Park Heating
#As it can seen that the Driving Range of the vehicle increases if there is a Park Heating.  Hen
ce, this variable will be considered for the model.
Range_ParkHeating= df1 %>%
  group_by(park_heating) %>%
  summarise(avg_range = mean(`trip_distance(km)`)) %>%
  arrange(desc(avg_range))

DT::datatable(Range_ParkHeating)
```

Show 10 entries Search:

| | park_heating | avg_range |
|---|---|---|
| 1 | 1 | 46.7489451476793 |
| 2 | 0 | 26.742033077854 |

Showing 1 to 2 of 2 entries Previous 1 Next

```
plt_parkheating = df1 %>% group_by(park_heating) %>% summarise(avg_range = mean(`trip_distance(km)`)) %>%
  ggplot(aes(x=park_heating,y=avg_range)) + geom_bar(stat="identity") +
  labs(title = "Average Driving Range by park_heating")
plot(plt_parkheating)
```




```
### Correlation overview-
# As seen in Correlation Chart and Scatter plots below, quantity(kWh) and avg_speed(km/h) seem to have too much correlation with the Target variable.  Hence, these 2 variables will be considered for the model.  The other 2 numerical variables such as "consumption(kWh/100km)" and "ecr_deviation" dont have any correlation, hence will be removed.
library(correlation)
```

```
## Warning: package 'correlation' was built under R version 4.0.5
```

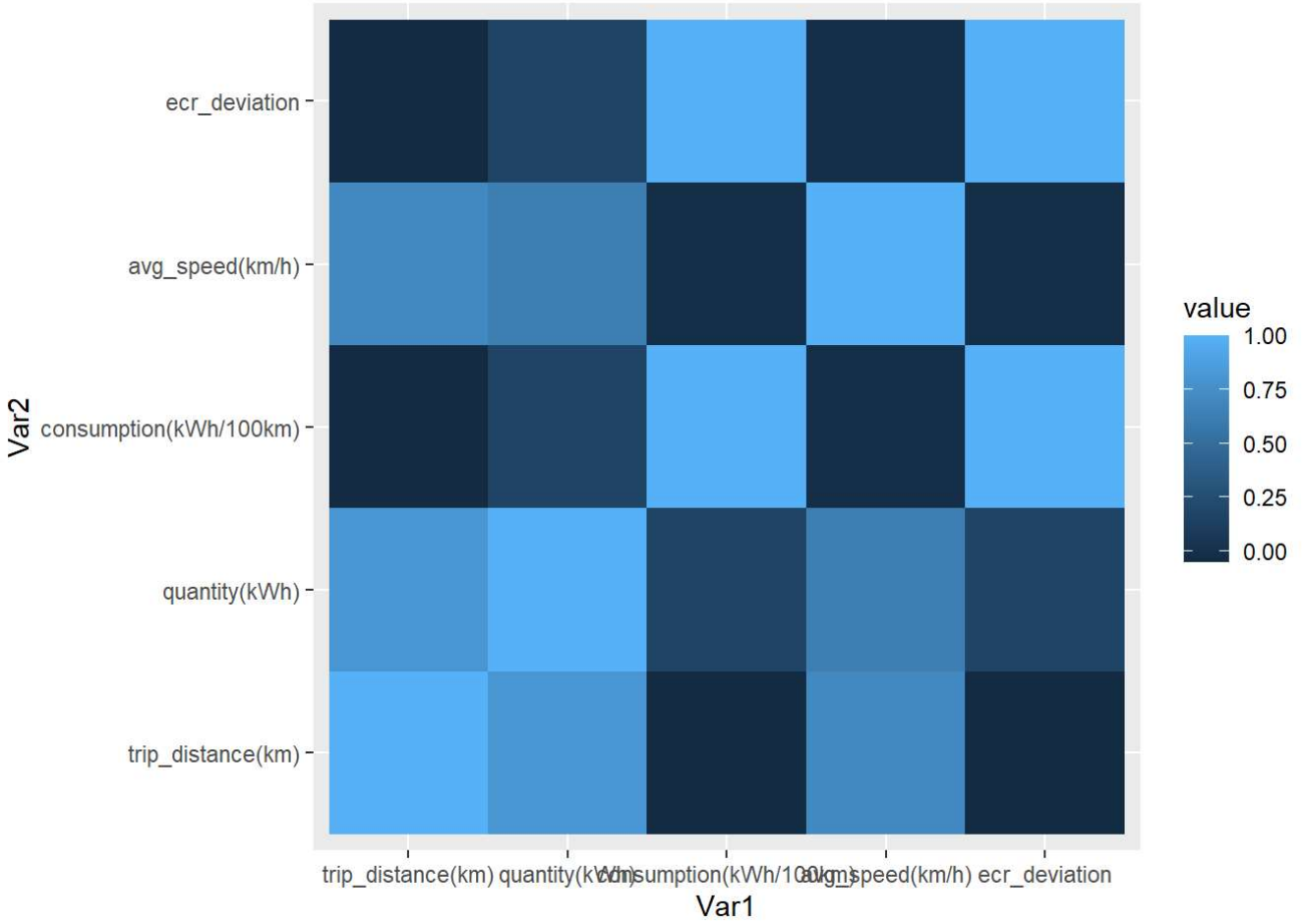
```
num_col<-sapply(df1, is.numeric)
df_num_cols<-df1[, num_col]
cor(df_num_cols)
```

```
##                       trip_distance(km) quantity(kWh) consumption(kWh/100km)
## trip_distance(km)             1.00000000     0.8087108            -0.04876906
## quantity(kWh)                 0.80871076     1.0000000             0.16913134
## consumption(kWh/100km)       -0.04876906     0.1691313             1.00000000
## avg_speed(km/h)               0.70909194     0.6307378            -0.01267947
## ecr_deviation                -0.04876906     0.1691313             1.00000000
##                       avg_speed(km/h) ecr_deviation
## trip_distance(km)          0.70909194   -0.04876906
## quantity(kWh)              0.63073784    0.16913134
## consumption(kWh/100km)    -0.01267947    1.00000000
## avg_speed(km/h)            1.00000000   -0.01267947
## ecr_deviation             -0.01267947    1.00000000
```

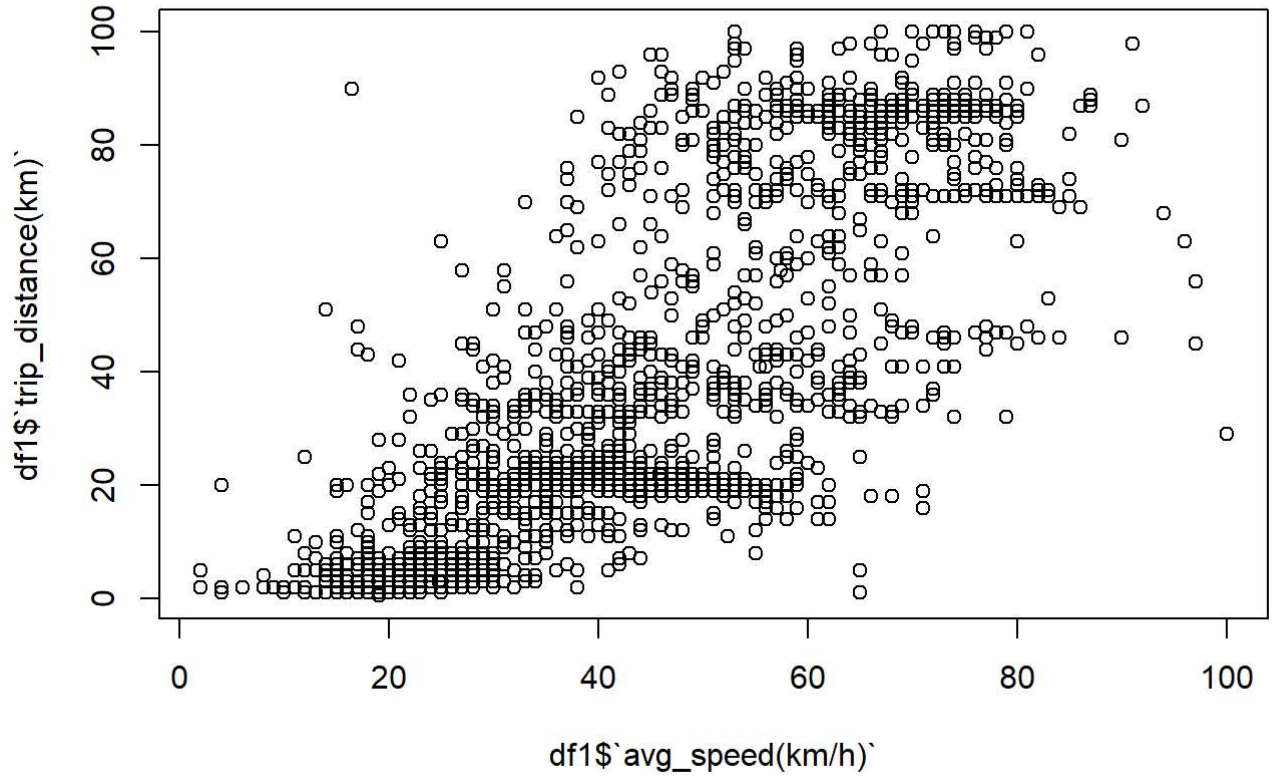
```
# Correlation Heat Chart
cormat <- round(cor(df_num_cols),3)
melted_cormat <- melt(cormat)
```

```
## Warning in melt(cormat): The melt generic in data.table has been passed a matrix
## and will attempt to redirect to the relevant reshape2 method; please note that
## reshape2 is deprecated, and this redirection is now deprecated as well. To
## continue using melt methods from reshape2 while both libraries are attached,
## e.g. melt.list, you can prepend the namespace like reshape2::melt(cormat). In
## the next version, this warning will become an error.
```

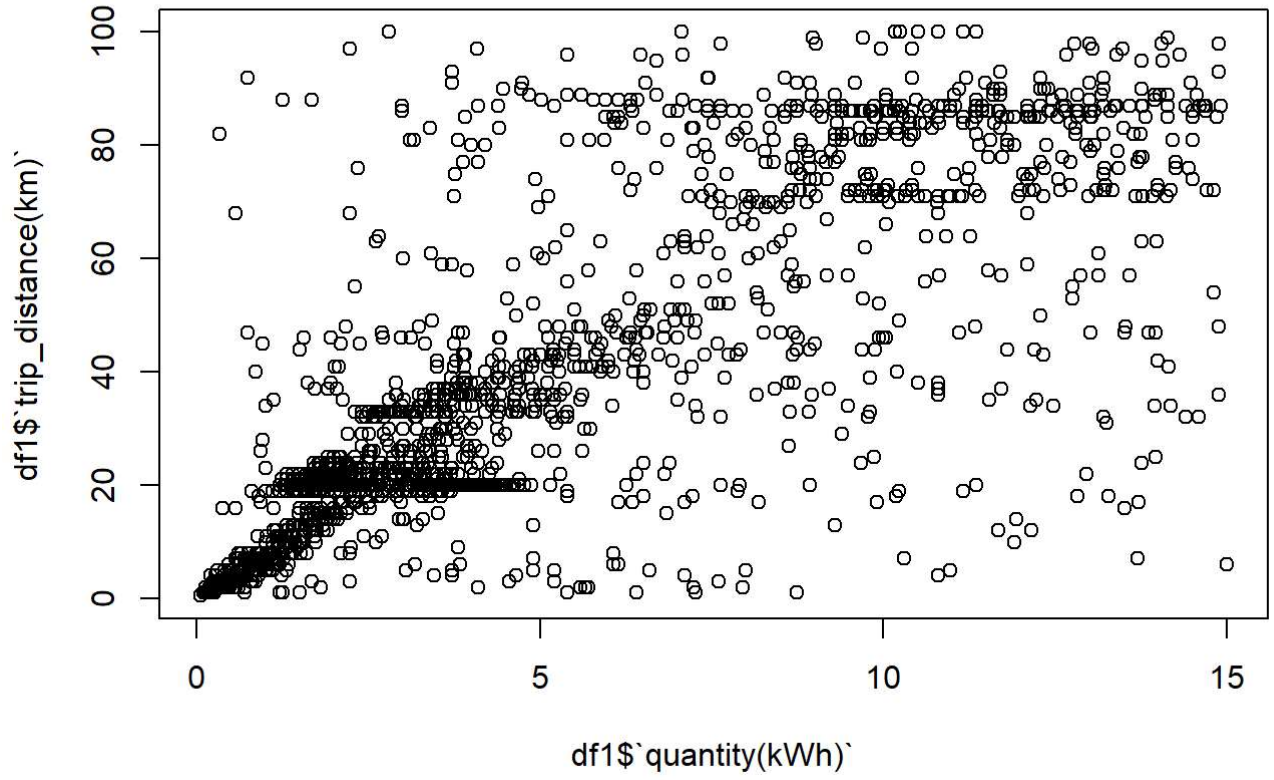
```
ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile()
```



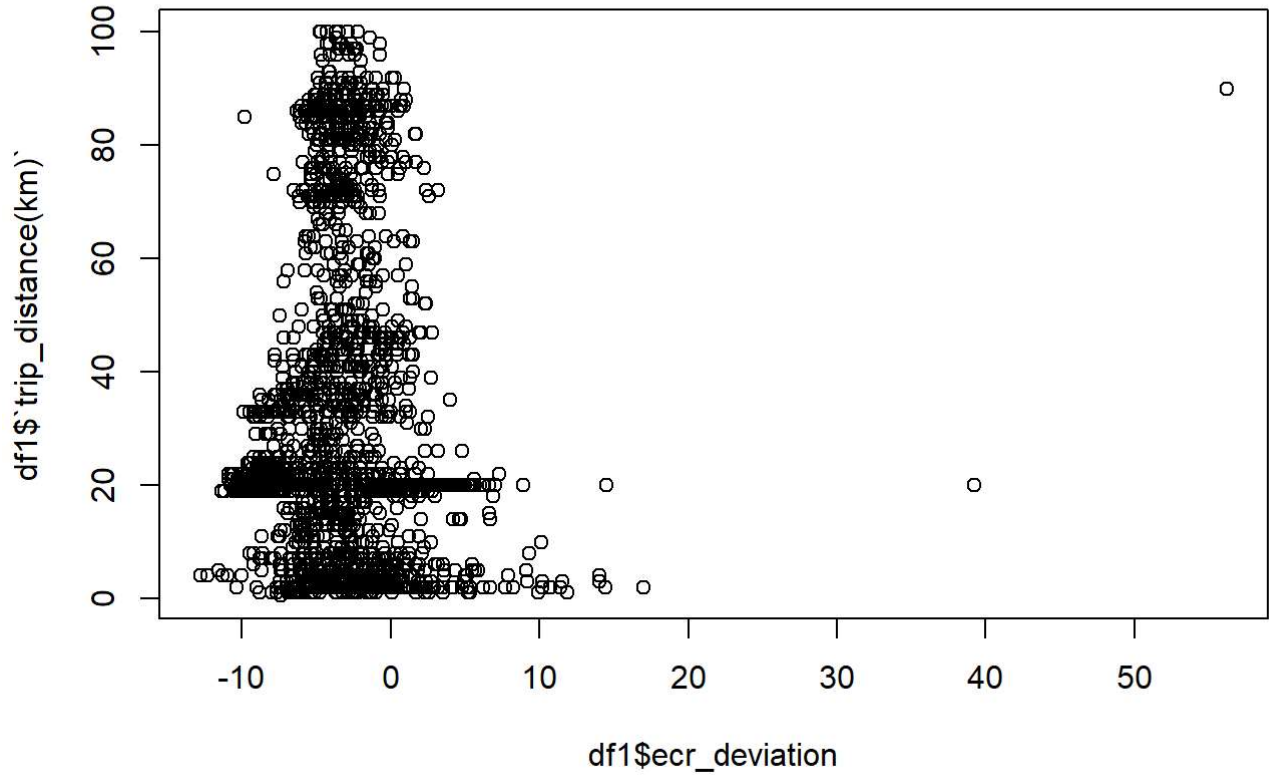

```
#Scatter Plots
plot(df1$`avg_speed(km/h)`,df1$`trip_distance(km)`)
```



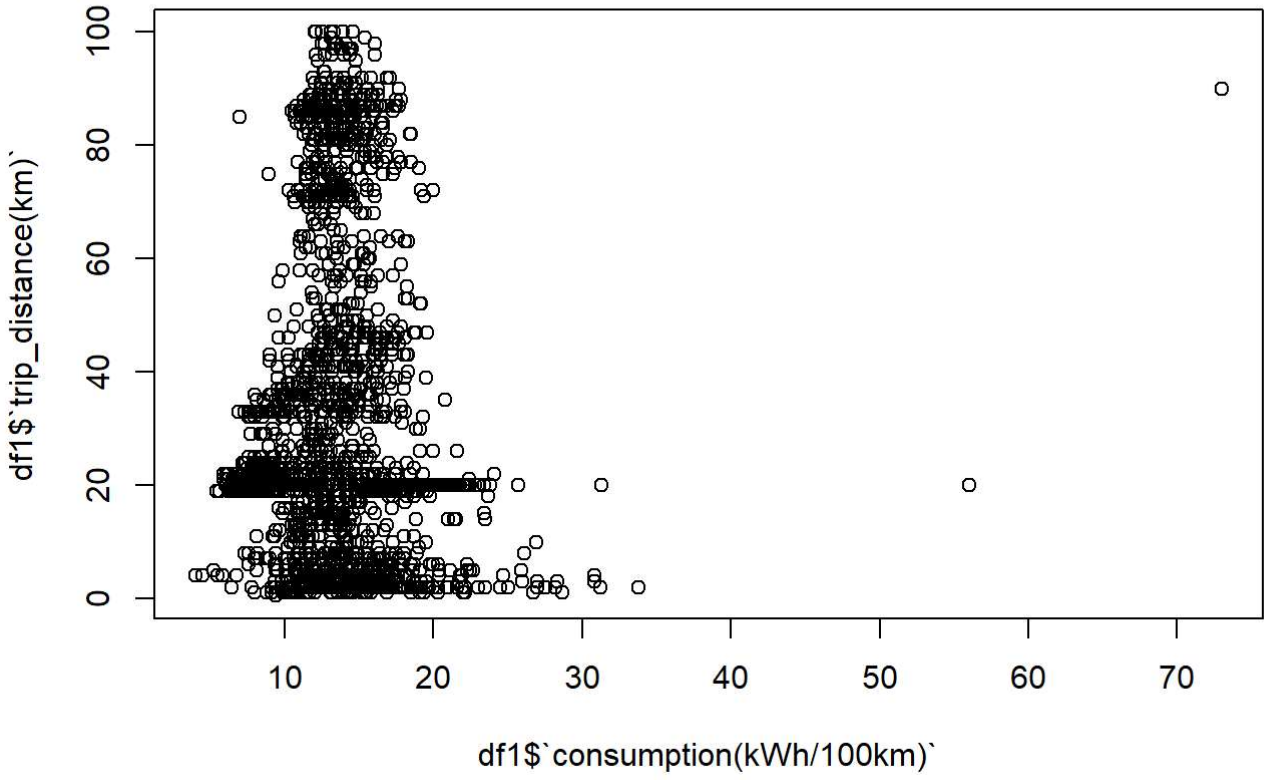

```
plot(df1$`quantity(kWh)`,df1$`trip_distance(km)`)
```




```
plot(df1$ecr_deviation,df1$`trip_distance(km)`)
```




```
plot(df1$`consumption(kWh/100km)`,df1$`trip_distance(km)`)
```




Hence, from the above Uni and Bi Variate Analysis, the following Variables will be Dropped from the model as they have very little or NO significance to predict the Dependent variable “trip_distance(km)” tire_type consumption(kWh/100km) A/C ecr_deviation

Removing the 4 variables from the dataframe. Now the Data is ready to build a predictive Model.

```
df_train<-within(df1, rm(tire_type, `A/C`, `consumption(kWh/100km)`, ecr_deviation))
```

LINEAR REGRESSION MODEL

```
#From the Model Output, Multiple R-squared:  0.7508 meaning, it is only 75% accuracte.  Lets try the Random Forest Model
dr_range<-lm(`trip_distance(km)`~`quantity(kWh)` + city + motor_way + country_roads + driving_style + park_heating + `avg_speed(km/h)`, data = df_train)
summary(dr_range)
```

```
##
## Call:
## lm(formula = `trip_distance(km)` ~ `quantity(kWh)` + city + motor_way +
##     country_roads + driving_style + park_heating + `avg_speed(km/h)`,
##     data = df_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -65.785  -5.337  -1.210   4.769  60.762
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              2.296888   1.924529   1.193  0.23278
## `quantity(kWh)`          3.404339   0.099006  34.385  < 2e-16 ***
## city1                  -13.839055   0.911389 -15.185  < 2e-16 ***
## motor_way1              -3.464944   0.585158  -5.921 3.56e-09 ***
## country_roads1          -0.004634   0.562093  -0.008  0.99342
## driving_styleModerate    4.598693   1.412089   3.257  0.00114 **
## driving_styleNormal      4.211156   1.444303   2.916  0.00358 **
## park_heating1           -1.351352   0.695098  -1.944  0.05198 .
## `avg_speed(km/h)`        0.503802   0.023933  21.051  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.58 on 2944 degrees of freedom
## Multiple R-squared:  0.7508, Adjusted R-squared:  0.7501
## F-statistic:  1108 on 8 and 2944 DF,  p-value: < 2.2e-16
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.5
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

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

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

```
library(caTools)
```

```
## Warning: package 'caTools' was built under R version 4.0.5
```

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.0.5
```

```
# Splitting data in train and test data
split <- sample.split(df_train, SplitRatio = 0.7)
split
```

```
## [1]  TRUE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE
```

```
train <- subset(df_train, split == "TRUE")
test <- subset(df_train, split == "FALSE")

dim(train)
```

```
## [1] 1846    8
```

```
dim(test)
```

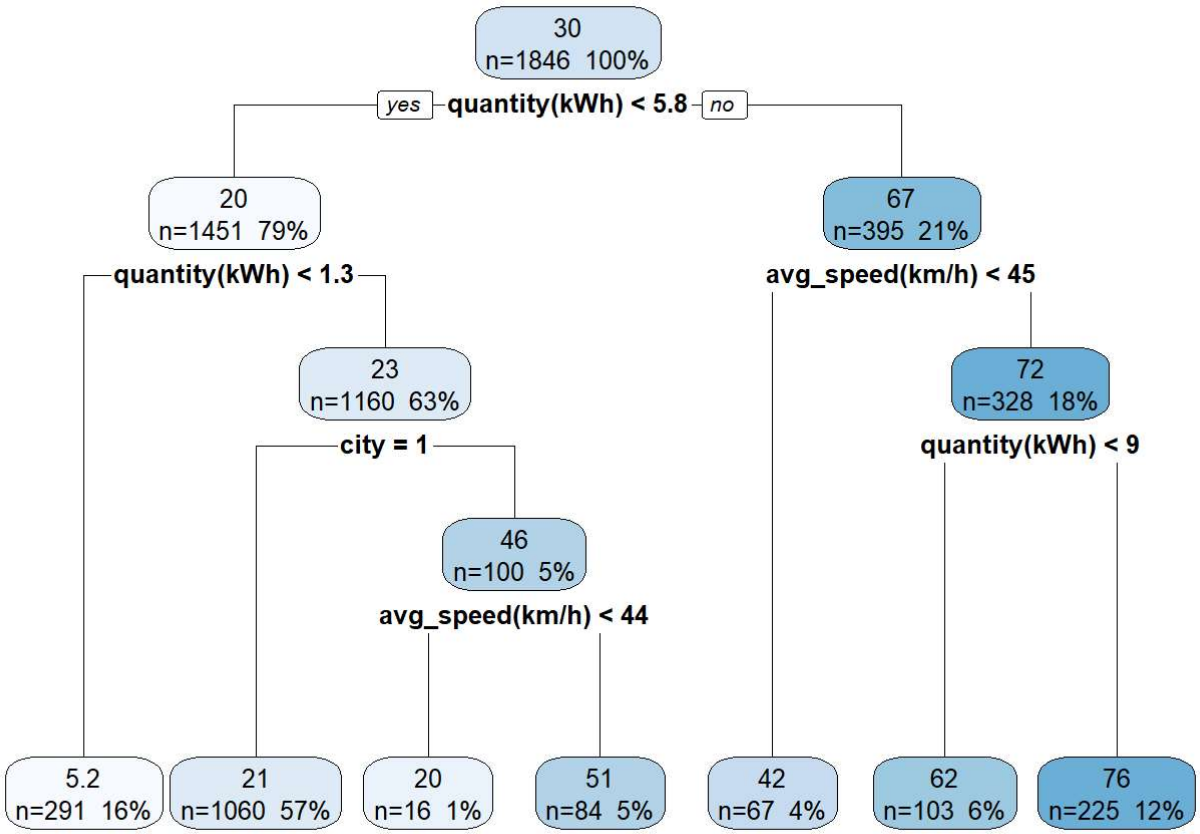
```
## [1] 1107    8
```

## R-PART MODEL

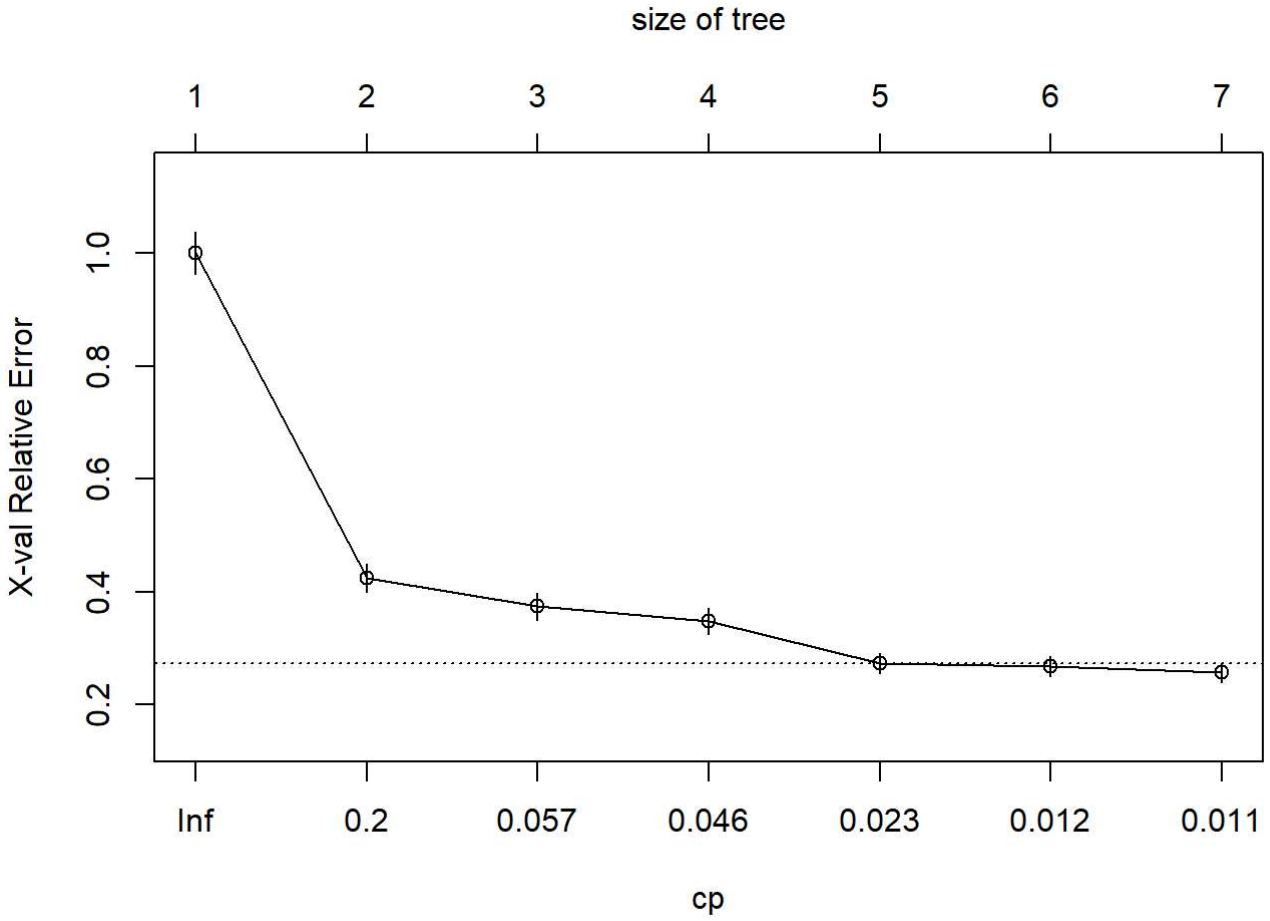
```
fit <- rpart(`trip_distance(km)`~., data = train, method = 'anova')
fit
```

```
## n= 1846
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 1846 1170230.000 29.727790
##    2) quantity(kWh)< 5.765 1451  251804.900 19.654380
##      4) quantity(kWh)< 1.275 291   19741.490  5.201031 *
##      5) quantity(kWh)>=1.275 1160  156023.900 23.280170
##       10) city=1 1060   34630.600 21.099060 *
##       11) city=0 100   62898.000 46.400000
##         22) avg_speed(km/h)< 43.5 16    1956.938 19.937500 *
##         23) avg_speed(km/h)>=43.5 84   47602.700 51.440480 *
##    3) quantity(kWh)>=5.765 395  230319.600 66.731650
##      6) avg_speed(km/h)< 44.5 67   40888.420 41.686570 *
##      7) avg_speed(km/h)>=44.5 328  138820.400 71.847560
##       14) quantity(kWh)< 8.96 103   52580.990 61.990290 *
##       15) quantity(kWh)>=8.96 225   71649.840 76.360000 *
```

```
rpart.plot(fit, extra=101)
```




```
plotcp(fit)
```




```
predict_rpart <-predict(fit, test)
predict_rpart
```

```
##         3         4         6        13        14        16        22        23
## 19.937500 76.360000 61.990291 19.937500 76.360000 76.360000 61.990291 21.099057
##        25        30        31        33        39        40        42        47
## 41.686567 21.099057 41.686567 51.440476 41.686567 61.990291 76.360000 61.990291
##        48        50        56        57        59        64        65        68
## 51.440476 51.440476 19.937500 76.360000 76.360000 41.686567 41.686567 76.360000
##        74        76        78        84        85        87        92        93
## 41.686567 61.990291 76.360000 76.360000 76.360000 21.099057 51.440476 61.990291
##        95       103       105       107       115       116       118       126
## 76.360000 76.360000 76.360000 51.440476 76.360000 76.360000 51.440476 76.360000
##       127       129       136       137       139       145       146       149
## 61.990291 41.686567 76.360000 51.440476 41.686567 76.360000 61.990291 76.360000
##       154       155       157       164       165       167       174       175
## 76.360000  5.201031 41.686567 76.360000 61.990291 61.990291 76.360000 76.360000
##       177       182       183       185       193       194       196       204
## 76.360000 76.360000 76.360000 76.360000 76.360000 61.990291 61.990291 76.360000
##       206       209       214       215       218       226       227       230
## 76.360000 76.360000  5.201031 76.360000 61.990291 61.990291 76.360000 76.360000
##       237       238       240       245       246       248       255       256
## 76.360000 19.937500 76.360000 61.990291  5.201031  5.201031 76.360000 76.360000
##       259       264       265       267       273       275       277       285
##  5.201031 51.440476 76.360000 76.360000 76.360000 61.990291  5.201031 76.360000
##       286       288       294       295       297       303       304       309
## 76.360000 76.360000 76.360000 51.440476 76.360000 76.360000 76.360000 19.937500
##       317       318       322       327       328       330       336       337
## 51.440476 76.360000 21.099057  5.201031 76.360000 76.360000 76.360000 76.360000
##       339       344       345       348       353       354       356       362
## 76.360000 61.990291 41.686567 76.360000 76.360000 61.990291  5.201031  5.201031
##       363       366       371       372       374       380       381       383
## 51.440476 76.360000 61.990291 76.360000 76.360000 76.360000 76.360000  5.201031
##       388       390       393       400       402       404       409       410
## 61.990291 76.360000 76.360000 51.440476 76.360000 21.099057 76.360000 61.990291
##       414       420       422       426       433       434       436       444
## 61.990291 76.360000 76.360000 76.360000 76.360000 21.099057 61.990291 76.360000
##       447       450       459       460       463       470       472       475
## 76.360000 76.360000 19.937500 76.360000 41.686567 61.990291 61.990291 51.440476
##       484       485       487       492       493       495       502       503
## 41.686567 76.360000 76.360000 19.937500 61.990291 76.360000 76.360000 76.360000
##       506       512       514       517       524       525       527       534
## 41.686567 51.440476 76.360000 51.440476 51.440476 76.360000 61.990291 61.990291
##       535       540       552       554       557       565       566       571
## 51.440476 76.360000 41.686567  5.201031 41.686567 76.360000 61.990291 76.360000
##       589       591       593       600       602       606       614       617
## 76.360000 21.099057 61.990291 76.360000 61.990291 76.360000 51.440476 76.360000
##       620       628       629       631       641       642       646       656
## 51.440476 21.099057 21.099057 76.360000 19.937500 76.360000 76.360000 51.440476
##       657       661       667       668       671       677       678       681
## 61.990291 76.360000 76.360000 61.990291 51.440476 76.360000 51.440476 61.990291
##       686       689       691       698       699       702       708       710
## 76.360000 41.686567 51.440476  5.201031 76.360000 76.360000 76.360000 41.686567
##       715       722       724       726       732       733       737       745
## 19.937500 61.990291 61.990291 51.440476 76.360000 76.360000 76.360000 76.360000
```

```
##       747       750       756       758       762       768       769       771
## 41.686567 51.440476 61.990291 21.099057 76.360000 76.360000 21.099057 61.990291
##       776       777       780       787       788       791       797       799
## 41.686567 61.990291 51.440476 76.360000 51.440476 41.686567 76.360000 61.990291
##       802       814       815       817       823       825       828       835
## 41.686567 61.990291 21.099057 41.686567 61.990291 51.440476 76.360000 61.990291
##       837       841       846       847       855       861       862       864
## 76.360000 61.990291 21.099057  5.201031 51.440476 41.686567 61.990291  5.201031
##       876       877       879       888       891       894       899       900
## 76.360000 19.937500 41.686567 76.360000 76.360000 76.360000 76.360000 51.440476
##       904       909       910       912       919       920       922       928
## 41.686567 51.440476 61.990291 61.990291 51.440476 19.937500 19.937500 76.360000
##       929       931       937       938       943       950       951       953
## 76.360000 21.099057 51.440476 76.360000 76.360000 76.360000  5.201031  5.201031
##       958       959       961       966       967       969       974       975
##  5.201031  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##       977       982       983       985       990       991       993       998
##  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##       999      1001      1006      1007      1009      1016      1017      1019
##  5.201031  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1024      1025      1027      1032      1033      1035      1042      1043
## 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057
##      1045      1051      1052      1054      1060      1061      1063      1068
## 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057  5.201031 21.099057
##      1069      1071      1077      1078      1081      1086      1087      1089
## 21.099057 21.099057 76.360000 21.099057 21.099057 21.099057 21.099057 21.099057
##      1094      1097      1099      1104      1105      1107      1114      1115
## 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057
##      1118      1124      1125      1127      1132      1133      1135      1140
##  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1141      1143      1148      1149      1151      1156      1157      1159
## 76.360000 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1164      1165      1167      1174      1175      1177      1182      1186
## 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057 76.360000 76.360000
##      1188      1193      1194      1196      1201      1202      1204      1209
## 21.099057 21.099057  5.201031 21.099057  5.201031 21.099057 21.099057 21.099057
##      1210      1212      1217      1218      1220      1225      1226      1228
##  5.201031 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057  5.201031
##      1233      1234      1236      1241      1242      1244      1251      1252
##  5.201031 21.099057  5.201031 21.099057  5.201031 21.099057 21.099057 21.099057
##      1254      1259      1260      1262      1267      1268      1271      1277
## 21.099057  5.201031  5.201031 21.099057 21.099057 21.099057 21.099057  5.201031
##      1278      1280      1285      1286      1292      1299      1300      1303
## 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 61.990291 21.099057
##      1308      1309      1311      1316      1317      1319      1324      1326
## 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 61.990291
##      1328      1333      1334      1336      1341      1342      1344      1349
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1350      1352      1357      1358      1360      1365      1366      1368
##  5.201031  5.201031 21.099057 21.099057  5.201031 21.099057 21.099057 61.990291
##      1373      1374      1376      1381      1382      1384      1389      1392
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
```

```
##      1394      1399      1400      1402      1407      1408      1410      1415
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1416      1418      1423      1424      1426      1433      1434      1436
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1443      1444      1446      1452      1455      1457      1462      1463
## 21.099057  5.201031 61.990291  5.201031 61.990291 61.990291 21.099057 21.099057
##      1465      1470      1471      1473      1478      1479      1481      1486
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057
##      1487      1489      1494      1496      1498      1504      1505      1507
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1512      1513      1515      1520      1521      1523      1528      1529
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1532      1538      1539      1541      1546      1547      1549      1554
##  5.201031 21.099057 21.099057 21.099057  5.201031  5.201031 21.099057 21.099057
##      1555      1557      1562      1563      1565      1570      1571      1573
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1581      1582      1584      1589      1590      1592      1597      1598
## 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057
##      1600      1607      1608      1610      1615      1616      1618      1623
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 76.360000 21.099057
##      1624      1626      1634      1635      1637      1642      1643      1645
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1650      1651      1653      1658      1659      1661      1666      1667
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 61.990291 21.099057
##      1669      1674      1675      1677      1683      1684      1686      1691
##  5.201031 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057
##      1692      1694      1700      1701      1703      1708      1709      1711
## 21.099057  5.201031 21.099057 21.099057 21.099057  5.201031  5.201031 21.099057
##      1716      1717      1719      1725      1726      1728      1733      1734
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1736      1742      1743      1745      1751      1752      1754      1759
##  5.201031 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057
##      1760      1764      1769      1770      1772      1778      1779      1781
## 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1786      1787      1789      1794      1795      1797      1804      1805
## 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057
##      1807      1815      1816      1818      1823      1824      1826      1831
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057
##      1832      1834      1841      1842      1844      1849      1850      1852
## 21.099057 21.099057 61.990291 21.099057 21.099057  5.201031 21.099057 21.099057
##      1858      1859      1861      1866      1867      1870      1875      1876
## 21.099057 21.099057 21.099057 21.099057 21.099057 41.686567 21.099057 21.099057
##      1878      1885      1886      1888      1893      1894      1896      1901
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057
##      1902      1904      1909      1910      1912      1917      1918      1920
## 21.099057 21.099057  5.201031  5.201031 21.099057 21.099057  5.201031 21.099057
##      1925      1926      1928      1933      1934      1936      1941      1942
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      1944      1949      1950      1952      1957      1958      1960      1965
## 21.099057 21.099057 61.990291 21.099057 21.099057  5.201031  5.201031 21.099057
##      1966      1968      1973      1974      1976      1982      1983      1985
## 21.099057 21.099057 61.990291 21.099057 61.990291 21.099057 21.099057 21.099057
```

```
##      1990      1991      1993      1998      1999      2001      2007      2008
##  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057
##      2010      2015      2016      2018      2026      2027      2029      2036
## 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057
##      2037      2039      2044      2045      2047      2052      2053      2055
## 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057
##      2060      2061      2064      2070      2071      2073      2078      2079
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 76.360000
##      2082      2087      2088      2090      2095      2096      2098      2103
## 21.099057 21.099057  5.201031 21.099057 41.686567 21.099057 21.099057  5.201031
##      2104      2106      2112      2113      2117      2125      2126      2128
## 21.099057 21.099057 41.686567 41.686567  5.201031 21.099057 21.099057 21.099057
##      2133      2134      2136      2142      2143      2145      2150      2151
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      2153      2160      2161      2163      2168      2169      2171      2176
## 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057
##      2177      2179      2184      2185      2187      2192      2193      2195
## 21.099057 41.686567 21.099057 21.099057 21.099057 21.099057 19.937500 76.360000
##      2200      2201      2203      2209      2210      2213      2218      2219
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031
##      2221      2226      2227      2229      2234      2235      2237      2242
## 21.099057 21.099057 21.099057 21.099057 76.360000 61.990291 21.099057 21.099057
##      2243      2245      2250      2251      2253      2258      2259      2261
## 21.099057 21.099057 21.099057 61.990291 61.990291 21.099057 21.099057 21.099057
##      2266      2267      2269      2274      2275      2277      2282      2283
## 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057
##      2285      2290      2291      2293      2298      2299      2301      2306
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      2307      2311      2316      2317      2319      2325      2326      2328
## 19.937500 61.990291 21.099057  5.201031 21.099057  5.201031 21.099057 21.099057
##      2333      2334      2336      2343      2344      2346      2351      2352
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      2355      2360      2361      2363      2368      2369      2371      2376
## 61.990291 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057
##      2377      2379      2384      2385      2387      2392      2393      2395
## 51.440476 21.099057 61.990291  5.201031 21.099057 21.099057  5.201031 51.440476
##      2400      2401      2403      2408      2409      2411      2416      2417
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      2420      2425      2426      2428      2433      2434      2436      2441
## 21.099057 21.099057  5.201031 21.099057  5.201031 21.099057 21.099057  5.201031
##      2442      2444      2449      2450      2452      2457      2458      2460
## 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057  5.201031
##      2465      2467      2470      2475      2476      2478      2485      2486
## 21.099057  5.201031 21.099057 21.099057  5.201031 21.099057  5.201031 21.099057
##      2488      2493      2494      2496      2501      2502      2504      2510
##  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031
##      2511      2513      2518      2519      2521      2526      2527      2529
##  5.201031  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      2534      2535      2537      2542      2543      2545      2550      2551
## 21.099057 21.099057 21.099057 21.099057  5.201031 51.440476 21.099057 21.099057
##      2553      2558      2559      2561      2566      2567      2569      2574
## 21.099057 21.099057  5.201031  5.201031 21.099057 21.099057  5.201031 76.360000
```

```
##      2577      2579      2584      2585      2587      2592      2593      2595
## 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057
##      2600      2601      2603      2608      2609      2611      2616      2617
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 76.360000
##      2619      2624      2625      2627      2632      2633      2635      2640
## 21.099057  5.201031 21.099057 21.099057 21.099057 61.990291 21.099057 21.099057
##      2641      2643      2648      2649      2651      2656      2657      2659
##  5.201031 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057
##      2664      2665      2667      2672      2673      2675      2683      2684
##  5.201031  5.201031 41.686567 21.099057  5.201031 21.099057 21.099057 21.099057
##      2686      2691      2692      2694      2699      2700      2702      2707
## 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057
##      2708      2710      2715      2716      2718      2723      2724      2726
## 21.099057 21.099057 21.099057  5.201031  5.201031 21.099057 21.099057  5.201031
##      2731      2732      2734      2739      2740      2742      2747      2748
##  5.201031  5.201031  5.201031 21.099057 21.099057  5.201031 21.099057 21.099057
##      2750      2755      2756      2758      2763      2764      2766      2771
## 21.099057 76.360000 21.099057  5.201031 21.099057  5.201031 21.099057  5.201031
##      2772      2774      2779      2780      2782      2787      2788      2790
##  5.201031  5.201031 21.099057  5.201031 21.099057 61.990291  5.201031  5.201031
##      2795      2796      2798      2803      2804      2806      2811      2812
## 21.099057 21.099057 21.099057  5.201031  5.201031 21.099057  5.201031 21.099057
##      2814      2819      2820      2822      2827      2828      2830      2836
## 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 41.686567
##      2837      2839      2844      2845      2847      2852      2853      2855
## 19.937500 21.099057 21.099057 21.099057 21.099057 21.099057 41.686567  5.201031
##      2860      2861      2863      2868      2869      2871      2878      2879
## 21.099057 21.099057  5.201031 61.990291 21.099057  5.201031 21.099057 21.099057
##      2881      2887      2888      2891      2896      2897      2899      2904
## 21.099057  5.201031  5.201031 21.099057 21.099057  5.201031 21.099057  5.201031
##      2905      2907      2912      2913      2915      2920      2921      2923
##  5.201031  5.201031  5.201031 21.099057  5.201031 61.990291 21.099057  5.201031
##      2928      2929      2931      2936      2937      2939      2944      2945
##  5.201031  5.201031  5.201031 21.099057 21.099057 21.099057 21.099057 21.099057
##      2947      2952      2953      2955      2963      2964      2966      2971
## 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057
##      2972      2975      2980      2981      2983      2990      2991      2993
## 21.099057 21.099057 21.099057 21.099057 21.099057 76.360000 21.099057 21.099057
##      2998      2999      3001      3007      3008      3010      3015      3016
## 21.099057  5.201031 21.099057  5.201031 21.099057  5.201031 21.099057  5.201031
##      3018      3023      3024      3026      3031      3032      3034      3039
## 21.099057 61.990291  5.201031  5.201031 21.099057 21.099057 61.990291  5.201031
##      3040      3044      3049      3050      3053      3058      3059      3061
## 19.937500 21.099057 21.099057 21.099057 21.099057 21.099057  5.201031  5.201031
##      3067      3068      3070      3077      3078      3080      3085      3086
##  5.201031  5.201031  5.201031  5.201031  5.201031 51.440476  5.201031 21.099057
##      3088      3093      3094      3096      3101      3102      3104      3109
## 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057 21.099057 21.099057
##      3110      3112      3117      3118      3120      3126      3127      3129
## 21.099057  5.201031  5.201031  5.201031  5.201031 21.099057 21.099057  5.201031
##      3134      3135      3137      3142      3143      3145      3150      3151
## 21.099057  5.201031 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057
```

```
##      3153      3158      3159      3161      3166      3167      3170      3175
## 21.099057 51.440476 21.099057 21.099057  5.201031  5.201031 21.099057  5.201031
##      3176      3178      3183      3184      3186      3191      3192      3194
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      3200      3202      3204      3209      3210      3212      3218      3219
##  5.201031 21.099057 21.099057 21.099057 21.099057  5.201031 21.099057 21.099057
##      3221      3226      3227      3229      3234      3235      3237      3242
## 21.099057  5.201031  5.201031 21.099057  5.201031 21.099057 21.099057 21.099057
##      3243      3245      3251      3252      3254      3260      3261      3263
## 21.099057 21.099057 21.099057 21.099057  5.201031  5.201031 21.099057 21.099057
##      3268      3269      3271      3276      3277      3279      3284      3285
## 21.099057  5.201031 41.686567 21.099057  5.201031 21.099057 21.099057  5.201031
##      3287      3292      3293      3295      3300      3301      3303      3308
## 21.099057  5.201031 41.686567  5.201031  5.201031 21.099057 21.099057 21.099057
##      3309      3311      3316      3317      3319      3324      3327      3329
## 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057 21.099057
##      3334      3335      3337
## 21.099057 21.099057 41.686567
```

```
summary(predict_rpart)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.201  21.099  21.099  29.205  21.099  76.360
```

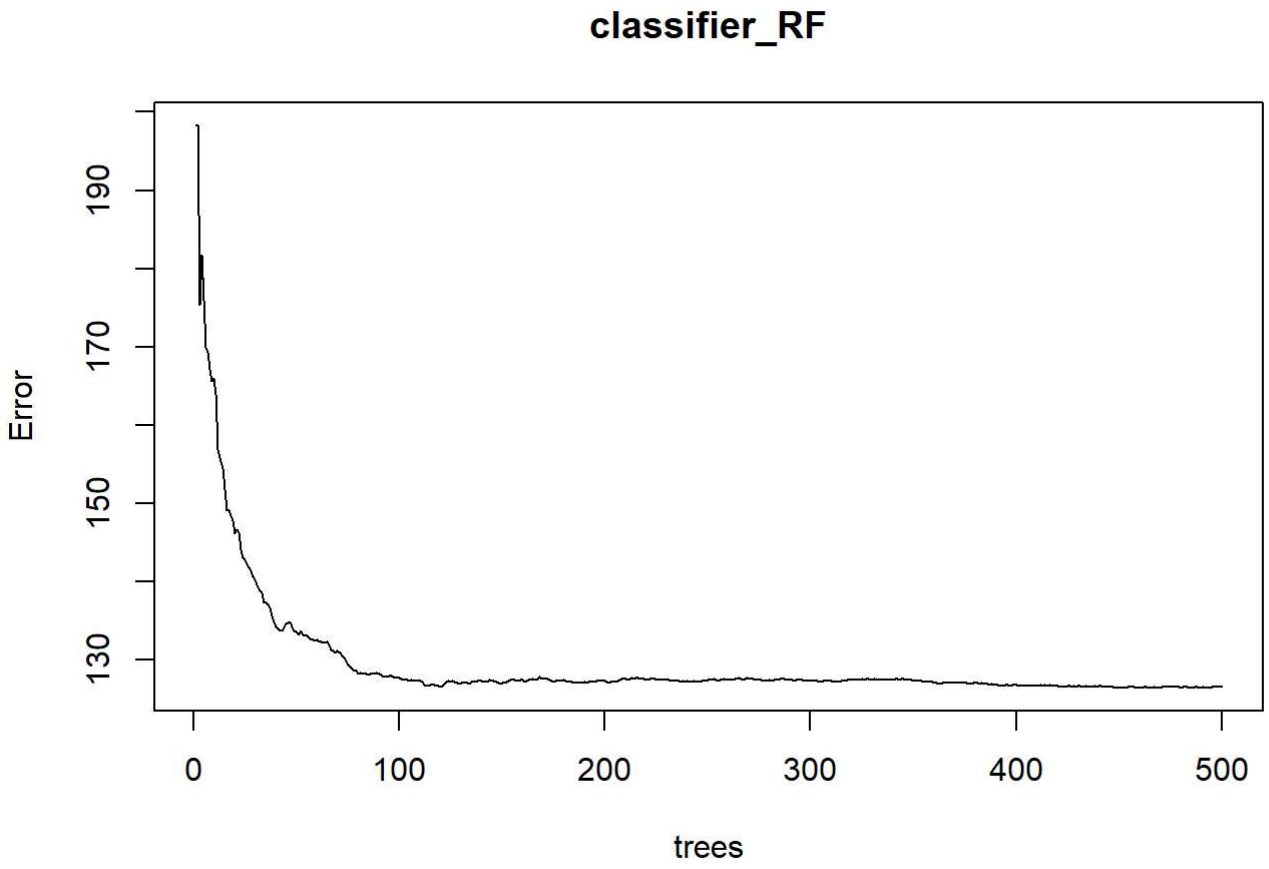
```
#Adding RPart Predicted values to the test dataframe
test$Rpart_Pred<-predict_rpart
```

## RANDOM FOREST MODEL

```
set.seed(120)  # Setting seed
classifier_RF = randomForest(x = train[-1],
                             y = train$`trip_distance(km)`,
                             ntree = 500)

print(classifier_RF)
```

```
##
## Call:
##  randomForest(x = train[-1], y = train$`trip_distance(km)`, ntree = 500)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##           Mean of squared residuals: 126.6029
##                     % Var explained: 80.03
```

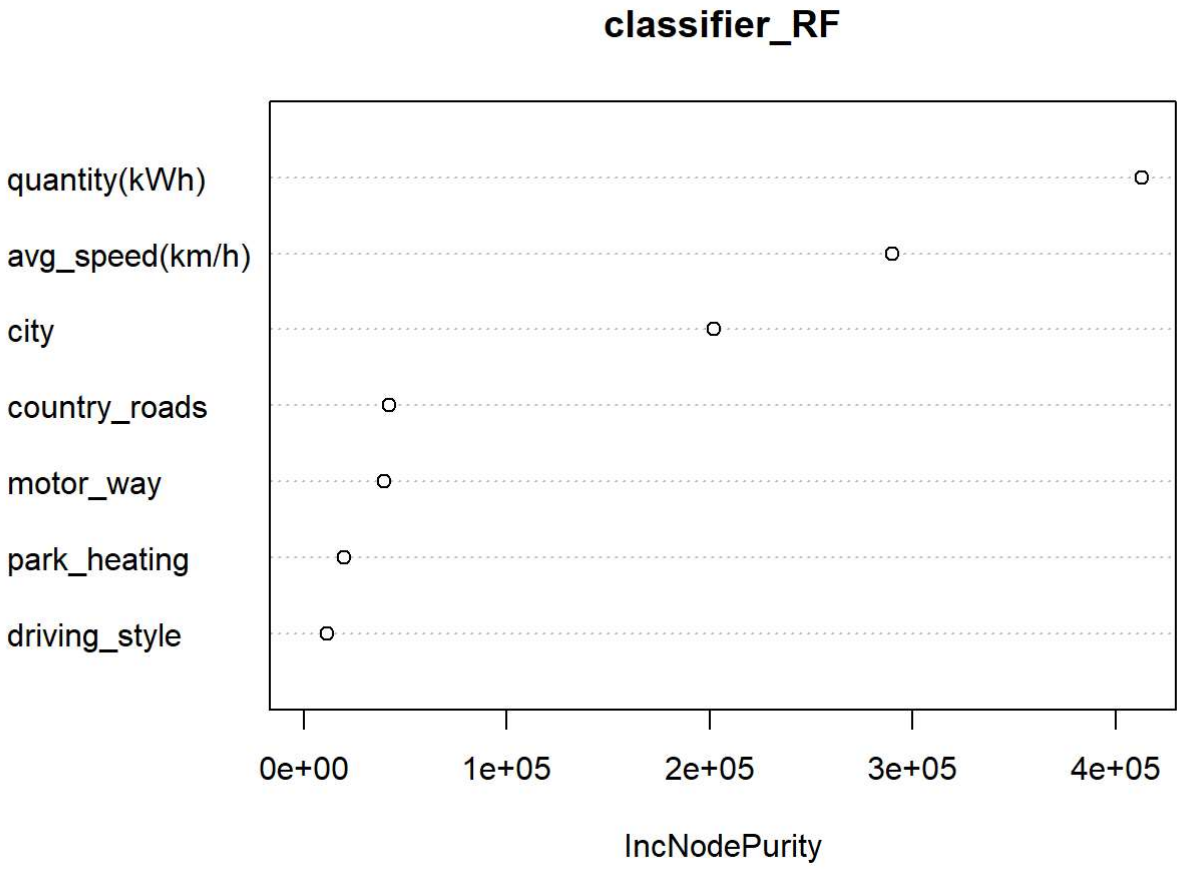
```
library(dplyr)
# Predicting the Test set results
y_pred = predict(classifier_RF, newdata = test[-1])

#Adding RF Predicted values to the test dataframe
test$RandomForest_Pred<-y_pred


# Plotting model
plot(classifier_RF)
```




```
# Importance plot
importance(classifier_RF)
```

```
##                 IncNodePurity
## quantity(kWh)       412961.07
## city                202194.15
## motor_way            39630.21
## country_roads        42186.58
## driving_style        11319.30
## park_heating         19878.51
## avg_speed(km/h)     290059.05
```

```
# Variable importance plot
varImpPlot(classifier_RF)
```




As seen from the above results, Random Forest model is performing better than LR or RPart models. The RF model is giving an accuracy of 81.47% which is descent enough. We can still increase the accuracy by looking at Deep Learning models or certain parameter tuning in the current RF Models.

Exporting the Predicted output data into Excel

```
library("writexl")
```

```
## Warning: package 'writexl' was built under R version 4.0.5
```

```
write_xlsx(test,"prediction_output.xlsx")
```

THANK YOU