Report

Project Phase II

Car Insurance Claim Prediction

Statistical Inference

Fatemeh Nadi

810101285

January, 2023

Contents

1 Question 1

Consider two categorical variables in your dataset for which at least one of them has more than two levels. Using these, follow these steps:

The first step is to determine which variables are categorical.

```
> # categorical variables
> car.categorical <- car %>% select_if(negate(is.numeric))
> colnames(car.categorical)
 [1] "policy_id"          "area_cluster"       "segment"            "model"              "fuel_type"
 [6] "engine_type"        "is_parking_sensors" "is_parking_camera"  "rear_brakes_type"   "transmission_type"
[11] "steering_type"      "is_speed_alert"
```

Figure 1: Categorical variables of our dataset

Columns with unique values are counted.

```
> # number of unique variables for each colums
> car.categorical.t <- as.data.frame(t(car.categorical))
> apply(car.categorical.t, 1, function(car.categorical) length(unique(car.categorical)))
         policy_id       area_cluster            segment              model          fuel_type        engine_type
             30000                 22                  6                 11                  3                 11
is_parking_sensors  is_parking_camera   rear_brakes_type  transmission_type      steering_type     is_speed_alert
                 2                  2                  2                  2                  3                  2
```

Figure 2: Number of unique value for each columns

Also, see the numerical variables:

```
> # numberical variables
> car.numeric <- car %>% select_if((is.numeric))
> colnames(car.numeric)
 [1] "X"                  "policy_tenure"      "age_of_car"         "age_of_policyholder" "population_density"
 [6] "airbags"            "displacement"       "cylinder"           "gear_box"            "turning_radius"
[11] "length"             "width"              "height"             "gross_weight"        "ncap_rating"
[16] "is_claim"
>
> # number of unique variables for each colums
> car.numeric.t <- as.data.frame(t(car.numeric))
> apply(car.numeric.t, 1, function(car.numeric) length(unique(car.numeric)))
                 X       policy_tenure          age_of_car age_of_policyholder  population_density
             30000               30000                  43                  71                  22
           airbags        displacement            cylinder            gear_box      turning_radius
                 3                   9                   2                   2                   9
            length               width              height        gross_weight         ncap_rating
                 9                  10                  11                  10                   5
          is_claim
                 2
```

Figure 3: For numerical variables, the number of unique values for each column

Several numerical variables have a limited number, so we can consider them categorical.

```
> # convert some types of numberical variables
> car$population_density <- as.factor(car$population_density)
> car$airbags <- as.factor(car$airbags)
> car$displacement <- as.factor(car$displacement)
> car$cylinder <- as.factor(car$cylinder)
> car$turning_radius <- as.factor(car$turning_radius)
> car$length <- as.factor(car$length)
> car$width <- as.factor(car$width)
> car$height <- as.factor(car$height)
> car$gross_weight <- as.factor(car$gross_weight)
> car$ncap_rating <- as.factor(car$ncap_rating)
```

Figure 4: Covert types

Again, see categorical variables in dataset:

```
> # categorical variables
> car.categorical <- car %>% select_if(negate(is.numeric))
> colnames(car.categorical)
 [1] "policy_id"          "area_cluster"       "population_density" "segment"            "model"
 [6] "fuel_type"          "engine_type"        "airbags"            "is_parking_sensors" "is_parking_camera"
[11] "rear_brakes_type"   "displacement"       "cylinder"           "transmission_type"  "gear_box"
[16] "steering_type"      "turning_radius"     "length"             "width"              "height"
[21] "gross_weight"       "is_speed_alert"     "ncap_rating"
>
> # number of unique variables for each columns
> car.categorical.t <- as.data.frame(t(car.categorical))
> apply(car.categorical.t, 1, function(car.categorical) length(unique(car.categorical)))
         policy_id       area_cluster population_density            segment              model          fuel_type
             30000                 22                 22                  6                 11                  3
       engine_type            airbags is_parking_sensors  is_parking_camera   rear_brakes_type       displacement
                11                  3                  2                  2                  2                  9
          cylinder  transmission_type          gear_box      steering_type     turning_radius             length
                 2                  2                  2                  3                  9                  9
             width             height       gross_weight     is_speed_alert        ncap_rating
                10                 11                 10                  2                  5
```

Figure 5: Categorical variables in Car dataset

sampling from original dataset:

```
car_sample <- sample_n(car.categorical, 1000)
```

It is essential that at least one of the categorical variables has more than two levels. And when we calculated the contingency table, each cell had at least 10 cases.

At first, we observe every possible two combinations of categorical variables that met the above conditions.
We briefly show some of the contingency tables.

```
n <- 2:21
for(i in n){
  x <- i+1
  m <- x:22
  for(j in m){
    print(paste0(colnames(car_sample)[i],", ",colnames(car_sample)[j])
   )
    print(table(car_sample[,i],car_sample[,j]))
  }
}
```

```
[1] "area_cluster, transmission_type"

     Automatic Manual
  C1         6     24
  C10       20     40
  C11        2     12
  C12        9     20
  C13       24     44
  C14       30     30
  C15        5      8
  C16        2      9
  C17        3     13
  C18        0      1
  C19        7      7
  C2        44     74
  C20        1      3
  C21        2      6
  C22        1      3
  C3        25     98
  C4         1      9
  C5        35     75
  C6         2      9
  C7         5     35
  C8       107    102
  C9        14     33
```

Figure 6: The contingency table of area cluster and transmission type

```
[1] "is_parking_camera, ncap_rating"

       0   2   3   4   5
  No  285 304   0   0  25
  Yes  44  56 245  32   9
```

Figure 7: The contingency table of is_parking_camera and ncap rating

```
[1] "fuel_type, rear_brakes_type"

        Disc Drum
CNG        0  352
Diesel   245   57
Petrol     0  346
```

Figure 8: The contingency table of fuel type and gross weight

For all two combinations of variables, we have at least one cell which contains 0.

Chosen Categorical Variable : is_parking_camera and ncap rating, and for meeting the above conditions, we must merge its last 3 columns.

```r
car_sample <- within(car_sample, {
  ncap_rating.new <- NA # need to initialize variable
  ncap_rating.new[ncap_rating == 1] <- "0"
  ncap_rating.new[ncap_rating == 2] <- "2"
  ncap_rating.new[ncap_rating >= 3] <- "more than 2 star"
} )
```

I chose is parking camera with 2 levels and ncap rating with 5 levels. and then merge 3 last columns of them.

```
> # show the new contingency table
> print(paste0(colnames(car_sample)[10],", ",colnames(car_sample)[23]))
[1] "is_parking_camera, ncap_rating.new"
> print(table(car_sample[,10],car_sample[,23]))

       0   2 more than 2
  No  285 304          25
  Yes  44  56         286
```

Figure 9: The contingency table of is_parking_camera and ncap rating, After Merging

Now conditions are met.

1.1 A

Derive a 95% confidence interval for the difference of these two variables and interpret it

First, the conditions must be verified.

1. Independence:
1.1. Within groups: random sample → assume that this dataset collected randomly, and $n < 10\%$ population → $1000 < 10\%$ all the population(30000).
1.2. Between groups: There is only one NCAP rating per car, and the vehicle may or may not be equipped with a parking camera.

2. Sample size / skew: samples should meet the success-failure condition (at least 10 successes and 10 failures).

$$\begin{cases} n_1 \times p_1 > 10, & n_1 \times p_1 > 10, & n_1 \times p_1 > 10 \\ n_2 \times p_2 > 10248, & n_2 \times p_2 > 10, & n_2 \times p_2 > 10 \end{cases}$$

$$\begin{cases} 588 \times 0.4524 = 266, & 588 \times 0.5068 = 298, & 588 \times 0.0408 = 24 \\ 412 \times 0.1165 = 248, & 48 \times 0.1699 = 70, & 412 \times 0.7136 = 294 \end{cases}$$

Each groups has at least 10 expected cases.
All is met, the normal model can be used for the point estimate of the difference in support, where p_1 corresponds to hasn't camera to use for parking and p_2 to use camera for parking:

$$Point\ Estimate = p_1 - p_2$$

$$SE = \sqrt{\frac{p_1\,(1-p_1)}{n_1} + \frac{p_2\,(1-p_2)}{n_2}}$$

$CI = 95\% \rightarrow > qnorm((1 - .95)/2, lower.tail = FALSE) = 1.96$

Calculate 95% confidence interval:

$$Point\ Estimate \pm z^* SE \rightarrow Point\ Estimate \pm ME$$

Approximate 95% CI for difference proportion:

```r
# contingency table of ncap_rating and is_parking_camera
contingency_table <- addmargins(table(is_parking_camera, ncap_rating))
contingency_table

# calculate proportions
contingency_table_p<-contingency_table
contingency_table_p[1,]=round((contingency_table[1,]/contingency_table
    [1,4]),4)
contingency_table_p[2,]=round((contingency_table[2,]/contingency_table
    [2,4]),4)
contingency_table_p[1:2,1:3]

#calculate confidence intervar for each ratings
n1 <- contingency_table[1,4]
n2 <- contingency_table[2,4]

for(i in 1:length(colnames(contingency_table_p[1:2,1:3]))){

  p1 <- contingency_table_p[1,i]
  p2 <- contingency_table_p[2,i]

  pointest <- p1-p2

  se = sqrt(((p1*(1-p1))/n1)+((p2*(1-p2))/n2))

  c <- 0.95
  zscore <- qnorm((1-c)/2,lower.tail = FALSE)

  me <- zscore * se

  lowerinterval <- round(pointest - me,3)
  upperinterval <- round(pointest + me,3)

  print(paste0("Cofidence Interval for ", colnames(contingency_table_p
    )[i]," start(s) : (",lowerinterval, ", ", upperinterval, ")"))
}
```

Approximate 95% CI for difference proportion for each columns:

```
[1] "Cofidence Interval for 0 start(s) : (0.3, 0.401)"
[1] "Cofidence Interval for 2 start(s) : (0.297, 0.403)"
[1] "Cofidence Interval for more than 2 start(s) : (-0.747, -0.654)"
```

Figure 10: Result: CI for each columns of ncap_rating about difference proportion of is_parking_camera

Conclusions:

We are 95% confidence that the difference proportion between cars with cameras for parking and cars without cameras for parking, where NCAP's rating is low - 0 stars - is between 0.3 and 0.401.

We are 95% confidence that the difference proportion between cars with cameras for parking and cars without cameras for parking, where NCAP's rating is medium - 2 stars - is between 0.297 and 0.403.

We are 95% confidence that the difference proportion between cars with cameras for parking and cars without cameras for parking, where NCAP's rating is high - more than 2 stars - is between -0.747 and -0.656.

1.2 B

By hypothesis testing, determine if the two variables are independent or not

To test the independence, I test hypothesis using chi squared test.

Setting the hypothesis:

$$H_0 : NCAP\ rating\ is\ independent\ of\ using\ camera\ for\ parking.$$
$$H_a : NCAP\ rating\ is\ dependent\ of\ using\ camera\ for\ parking.$$

Conditions for the Chi-square Test:

1. Independence:
1.1 random sample/assignment: cars randomly selected.
1.2 if sampling without replacement, $n < 10\%$ of population: $1000 < 30000$
1.3 each case only contributes to one cell in the table and non-paired.

2. Sample size: Each particular scenario (i.e. cell) must have at least 5 expected cases.

```
> # show the new contingency table
> print(paste0(colnames(car_sample)[10],", ",colnames(car_sample)[23]))
[1] "is_parking_camera, ncap_rating.new"
> print(table(car_sample[,10],car_sample[,23]))

       0   2 more than 2
  No  285 304          25
  Yes  44  56         286
```

Figure 11: The contingency table of is_parking_camera and ncap rating, After Merging

All conditions are met.

Observed Values:				
Observed	0 star	2 stars	more than 2	Total
n	285	304	25	614
$\hat{p}$	0.4524	0.5068	0.04	

Expected Values:				
Expected	0 star	2 stars	more than 2	Total
n	202	221	191	614
$\hat{p}$	0.36	0.32	0.32	

$$Expected = \frac{\#\ row\ total \times \#\ columns\ total}{\#\ table\ total}$$

Chi-square test statistic formula:

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

$df = (C-1) \times (R-1) = 2 - 1 \times 3 - 1 = 2$
$> \ pchisq(238.03, 2, lower.tail = FALSE) = 2.053252e - 52 < 0.05$

Reject $H_0 \rightarrow$ Cars equipped with camera for parking are not randomly rated with NCAP among these three options, and certain options are favored over others.

```
> # Perform a chi-squared test of independence on the contingency table
> chi_squared_test <- chisq.test(contingency_table)
>
> # Print the p-value of the test to determine if the two variables are independent
> print(chi_squared_test$p.value)
[1] 5.433479e-114
```

Figure 12: The chi squared test and pvalue as result

χ^2 Test Statistic:				
Types	0 star	2 stars	more than 2	Total
Observed - Expected	83	83	-166	2
$(Observed - Expected)^2$	6869	6889	25576	41334
$\frac{(Observed-Expected)^2}{Expected}$	34	31	133.9	199

$\chi^2 = 199$

2 Question 2

Choose a binary categorical variable and randomly select a small sample of your data (small sample size, e.g., n 15).
Then, perform a hypothesis test for the variable's success rate by means of the Simulation method

Chosen Binary Categorical Variable : transmission_type.

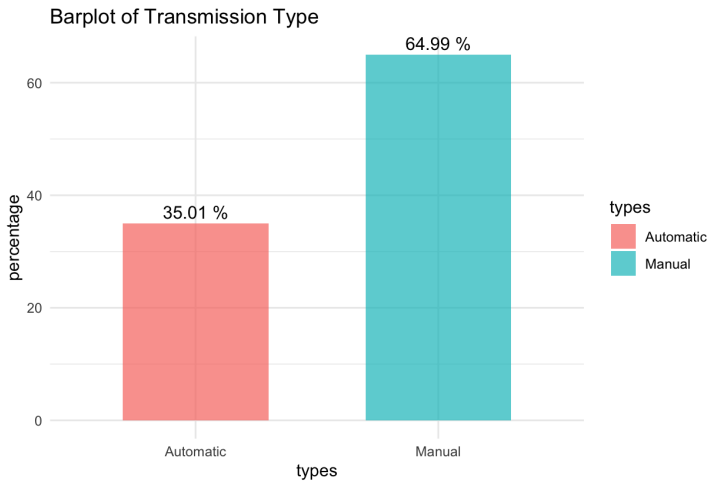

Figure 13: Barplot of transmission_type of popluation

sampling from original dataset:

```
car_sample <- sample_n(car.categorical, 15)
```

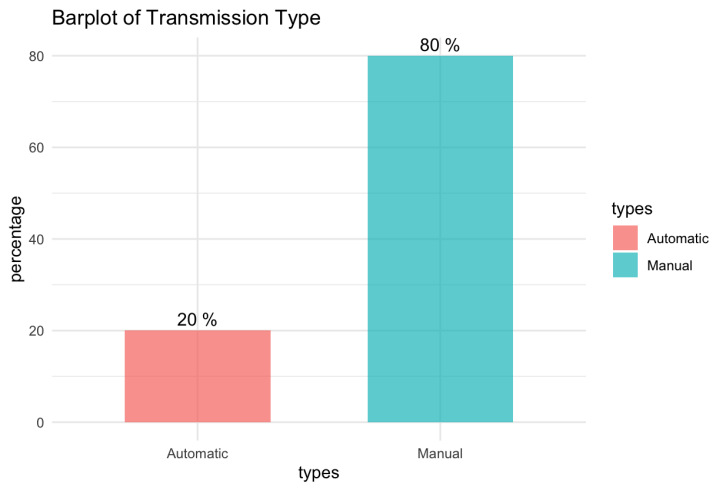

Figure 14: Barplot of transmission_type of sampling

Setting the hypothesis:

$$H_0 : half\ of\ the\ cars\ are\ manual\ and\ others\ are\ automatic\ \rightarrow p = 0.5.$$
$$H_a : greater\ than\ half\ of\ the\ cars\ are\ manual\ \rightarrow p > 0.5.$$

Checking conditions:
1. Independence: we can assume that cars are independent
2. Sample size / skew: $15 \times 0.5 = 7.5 < 10 \rightarrow$ not met.
distribution of sample proportions cannot be assumed to be nearly normal.

We will use simulation as a result of the fact that our conditions were not met.
To do this,
1. Use a fair coin, and label head as success (Manual).
2. One simulation: flip the coin 15 times and record the proportion of heads (Manual): $\hat{p}_{simulation}$.
3. Repeat the simulation many times (1000), recording the proportion of heads at each iteration: $\hat{p}_{simulation,1}, \hat{p}_{simulation,2}, ..., \hat{p}_{simulation.1000}$.
4. Calculate the percentage of simulations where the simulated proportion of heads is at least as extreme as the observed proportion ($\hat{p} = 0.8$).

For this simulation using 2 different methods :

Method 1: manually

```
transmission <- car_sample$transmission_type
p.hat <- table(transmission)[2]/15
transmission.simulation <-  data.frame(replicate(n = 1000, mean(sample
   (levels(as.factor(transmission)), size = 15, replace = TRUE)=='
   Manual')))
p_value <- mean(transmission.simulation >= p.hat)
p_value
```

```
> p_value <- mean(transmission.simulation >= p.hat)
> p_value
[1] 0.017
```

Figure 15: result of simulation - pvalue

Result: reject $H_0 \rightarrow$ greater than half of the cars are manual.

Method 2: Inference via Simulation

```
source("./inference.R")
inference(transmission,
    est="proportion",
    type="ht",
    success = "Manual",
          method = "simulation",
    null=0.5,
            alternative = "greater"
  )
```

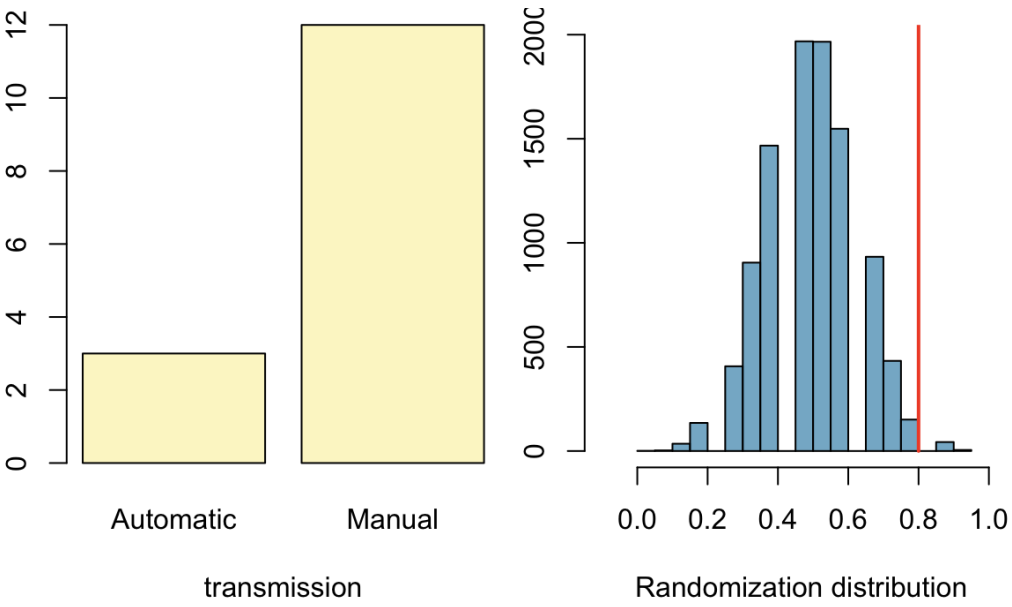

Figure 16: result of simulation via inference.R - barplot and histogram

```
> inference(transmission,est="proportion",
+           type="ht", success = "Manual",
+           method = "simulation",
+           null=0.5,
+           alternative = "greater")
Single proportion -- success: Manual
Summary statistics: p_hat = 0.8 ;  n = 15
H0: p = 0.5
HA: p > 0.5
p-value =  0.0199
```

Figure 17: result of simulation via inference.R - pvalue

Result: reject $H_0 \rightarrow$ greater than half of the cars are manual.

This comparison shows the same results from both methods, and both reject the null hypothesis.

3 Question 3

3.1 A

From your dataset, select two samples with a size of 100, plus a categorical variable with more than two levels.
One of the samples should be randomly selected and the other should be biased on purpose.
Compare each sample with the real distribution using 2 (goodness of fit) and interpret your results.

Chosen Categorical Variable : ncap_rating.

A random sample and a biased sample are taken, there is a bias in favor of five stars in the biased sample.

```
sample1 <- sample_n(car, 100)$ncap_rating
Number <- table(sample1)

# calculate proportions randomly sample
Proportion <- round((prop.table(Number)),3)
car.sample.unbiased <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.unbiased
```

```
> car.sample.unbiased
              0     2     3    4    5 Sum
Number    25.00 46.00 23.00 3.00 3.00 100
Proportion 0.25  0.46  0.23 0.03 0.03   1
```

Figure 18: The table of 100 samples of NCAP rating with proportions - randomly

```
pro <- ifelse(car$ncap_rating > 3 , 0.9, 0.1)
sample2 <- sample(car$ncap_rating, 100, prob = pro)
Number <- table(sample2)

# calculate proportions
Proportion <- round((prop.table(Number)),3)
car.sample.biased <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.biased
```

```
> car.sample.biased
              0     2     3     4     5 Sum
Number    20.0 21.00 17.00 15.00 27.00 100
Proportion 0.2  0.21  0.17  0.15  0.27   1
```

Figure 19: The table of 100 samples of NCAP rating with proportions - biased

The chi-square goodness of fit test is used to compare the observed distribution to an expected distribution, in a situation where we have two or more categories in a discrete data. In other words, it compares multiple observed proportions to expected probabilities.

Conditions for χ^2 test:
1. Independence:
1.1 random sample/assignment: cars randomly selected.
1.2 if sampling without replacement, $n < 10\%$ of population: $100 < 30000$
1.3 each case only contributes to one cell in the table and non-paired.

2. Sample size: Each particular scenario (i.e. cell) must have at least 5 expected cases.

```
> car.sample.unbiased
                0     2     3    4    5 Sum
Number      25.00 46.00 23.00 3.00 3.00 100
Proportion   0.25  0.46  0.23 0.03 0.03   1
> car.sample.biased
                0     2     3     4     5 Sum
Number      20.0 21.00 17.00 15.00 27.00 100
Proportion   0.2  0.21  0.17  0.15  0.27   1
```

Figure 20: random sample vs. biased sample

For meeting the above conditions, we must merge its last 2 columns.

```r
# merge 2 last columns
sample1.new <- ifelse(sample1 > 3, "more than 3", sample1)
sample2.new <- ifelse(sample2 > 3, "more than 3", sample2)

# sample1 : random sampling
Number <- table(sample1.new)
Proportion <- round((prop.table(Number)),3)
car.sample.unbiased.new <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.unbiased.new

# sample2 : biased sampling
Number <- table(sample2.new)
Proportion <- round((prop.table(Number)),3)
car.sample.biased.new <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.biased.new
```

```
> car.sample.unbiased.new
                0     2     3 more than 3 Sum
Number      25.00 46.00 23.00         6.00 100
Proportion   0.25  0.46  0.23         0.06   1
> car.sample.biased.new
                0     2     3 more than 3 Sum
Number      20.0 21.00 17.00        42.00 100
Proportion   0.2  0.21  0.17         0.42   1
```

Figure 21: new random sample vs. new biased sample

Now conditions are met.

In the first step, the χ^2 test was manually implemented to compare the distribution of biased and randomly selected sample.

Calculating expected value for NCAP rating from original population:

```
> pop <- car$ncap_rating
> p <- round(prop.table(table(ifelse(pop > 3, "more than 3", pop))),3)
> expected <- rbind(p*100,p)
> expected
       0      2      3 more than 3
  32.600 36.500 23.900        7.00
p  0.326  0.365  0.239        0.07
>
> df <- 4-1
```

Figure 22: Expected value from original population and degree of freedom

The Chi-square statistic is calculated as follow:

$$\chi^2 = \sum \frac{(o-e)^2}{e}$$

- o is the observed value.
- e is the expected value.

χ^2 test: random sampling

```
> chi.unbiased <- sum(((car.sample.unbiased.new[1:4]-expected[1:4])^2)/(expected[1:4]))
> chi.unbiased
[1] 4.286826
>
> pvalue.unbiased <- pchisq(chi.unbiased, df, lower.tail = FALSE)
> pvalue.unbiased
[1] 0.2321115
```

Figure 23: χ^2 test on random sampling and pvalue as a result

χ^2 test: random sampling

```
> chi.biased <- sum(((car.sample.biased.new[1:4]-expected[1:4])^2)/(expected[1:4]))
> chi.biased
[1] 11.56665
>
> pvalue.biased <- pchisq(chi.biased, df, lower.tail = FALSE)
> pvalue.biased
[1] 0.009025123
```

Figure 24: χ^2 test on biased sampling and pvalue as a result

Now, run χ^2 test by R function on random sampling and biased sampling.

```
> chisq.test(car.sample.unbiased.new[1,1:4], p = expected[2,1:4])

        Chi-squared test for given probabilities

data:  car.sample.unbiased.new[1, 1:4]
X-squared = 4.4211, df = 3, p-value = 0.2194

> chisq.test(car.sample.biased.new[1,1:4], p = expected[2,1:4])

        Chi-squared test for given probabilities

data:  car.sample.biased.new[1, 1:4]
X-squared = 188.44, df = 3, p-value < 2.2e-16
```

Figure 25: χ^2 test by R function

The results are the same for both methods (the R function and the manual implementation).

Conclusion:

For random sampling, the goodness of fit test has a high p-value.
Nevertheless, the p-value of the goodness of fit test for a biased sample is near zero, so we conclude that the biased sample has a different distribution from the population.
Therefore, both results match our expectations.

3.2 B

Pick up another categorical variable and compare it to the one you chose in part (a). Using the χ^2 test, check if the two variables are independent or not.

Select transmission type as categorical variable.

See contingency table:

```
> ncap_rating <- car$ncap_rating
> table(car$transmission_type,ncap_rating)
           ncap_rating
               0    2    3    4    5
  Automatic 1534 1797 7171    0    0
  Manual    8231 9154    0 1122  991
```

Figure 26: contingency table of NCAP rating and transmission type

Chi-square test examines whether rows and columns of a contingency table are statistically significantly associated. Setting the hypothesis:

H_0 :*the row and the column variables of the contingency table are independent.*
H_a :*row and column variables are dependent.*

Conditions for the Chi-square Test:

1. Independence:
1.1 random sample/assignment: cars randomly selected.
1.2 if sampling without replacement, $n < 10\%$ of population: $1000 < 30000$
1.3 each case only contributes to one cell in the table and non-paired.

2. Sample size: Each particular scenario (i.e. cell) must have at least 5 expected cases.

According to figure 3.2, to meet the conditions, we must merge the last three columns of the table.

```
# observed
ncap_rating.new <- ifelse(ncap_rating > 2, "more than 2", ncap_rating)
contingency.table <- addmargins(table(car$transmission_type,ncap_
    rating.new))
```

For each cell of the table, we have to calculate the expected value under null hypothesis.
For a given cell, the expected value is calculated as follow:

$$Expected = \frac{row.total * col.total}{table.total}$$

```r
# expected
expected<- contingency.table

numberOfcol <- ncol(expected)
numberOfrow <- nrow(expected)
table.total <- expected[numberOfrow,numberOfcol]

R <- 1:nrow(expected)
C <- 1:ncol(expected)

for (r in R) {
  for (c in C) {
    row.total   <- expected[r,numberOfcol]
    col.total   <- expected[numberOfrow,c]
    expected[r,c] <- round(row.total * col.total/ table.total)
  }
}
```

```
> contingency.table
            ncap_rating.new
                 0     2 more than 2   Sum
  Automatic   1534  1797        7171 10502
  Manual      8231  9154        2113 19498
  Sum         9765 10951        9284 30000
> expected
            ncap_rating.new
                 0     2 more than 2   Sum
  Automatic   3418  3834        3250 10502
  Manual      6347  7117        6034 19498
  Sum         9765 10951        9284 30000
```

Figure 27: Observed and Expected Value

The Chi-square statistic is calculated as follow:

$$\chi^2 = \sum \frac{(o-e)^2}{e}$$

- o is the observed value.
- e is the expected value.

This calculated Chi-square statistic is compared to the critical value (obtained from statistical tables) with $df = (r-1)*(c-1)$ degrees of freedom and $p = 0.05$.

- r is the number of rows in the contingency table.
- c is the number of column in the contingency table.

If the calculated Chi-square statistic is greater than the significance level, then we must conclude that the row and the column variables are not independent of each other. This implies that they are significantly associated.

```
> # chi square test
> chi <- sum(((contingency.table[1:2,1:3] - expected[1:2,1:3])^2)/(expected[1:2,1:3 ]))
> chi
[1] 10541.44
>
> df <- (length(R)-1) * (length(C)-1)
>
> pvalue <- pchisq(chi, 2, lower.tail = FALSE)
> pvalue
[1] 0
```

Figure 28: Chi square test and pvalue

Use R function to calculate chi square test and pvalue as result:

```
> # chi test with R functions
> chisq.test(contingency.table,  rescale.p = T)

        Pearson's Chi-squared test

data:  contingency.table
X-squared = 10541, df = 6, p-value < 2.2e-16
```

Figure 29: Chi square test and pvalue - R function

Reject $H_0 \rightarrow$ transmission type and NCAP rating are dependent.

4 Question 4

From your dataset choose a numerical variable that predicts its future value is meaningful within the context of your dataset. next, choose two explanatory variables which you believe are the best predictors for your response variable.

Chosen Numerical variable : policy_tenure as response variable and age_of_car and displacement as explanatory variables.

4.1 A

Without building a model yet, which explanatory variable do you guess is the more significant predictor and why? (use your knowledge from phase 1)

In phase I, we demonstrated that there are no strong discriminators in this dataset. Since the response variable(is_claim) is binary, I chose policy_tenure as a response variable, and the car's age and displacement as explanatory variables.
There are 21 numerical variables in this dataset, but only three are continuous, and the rest are discrete.

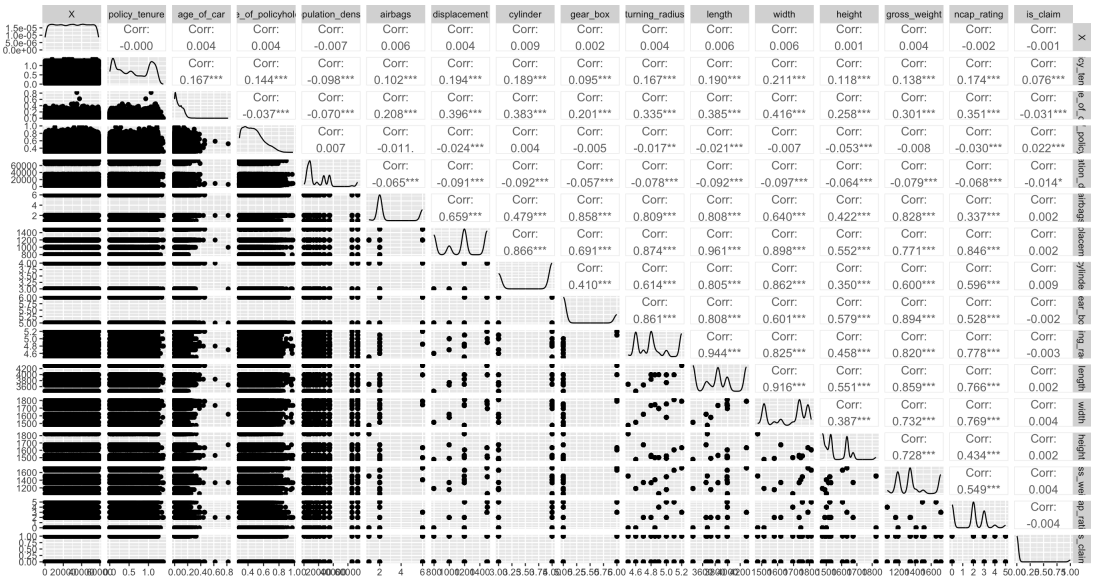

Figure 30: Correlogram phase I - numerical variable

Based on my knowledge from phase I, policy_tenure can be influenced by several variables, such as the car's age, model, and displacement.

4.2 B

First, build models:

```
model.with.age <- lm(car$policy_tenure ~ car$age_of_car)
model.with.dis <- lm(car$policy_tenure ~ car$displacement)
```

Conditions for linear regression :

1. Linear relationship: There exists a linear relationship between the independent variable, x, and the dependent variable, y.

2. Independence: The residuals are independent. In particular, there is no correlation between consecutive residuals in time series data.

3. Homoscedasticity: The residuals have constant variance at every level of x.

4. Normality: The residuals of the model are normally distributed.

for each explanatory variable:

1. age_of_car

4.2.1 a

Check the Linearity, Nearly Normal Residuals, and Constant Variability conditions in R

Method 1: use **library(ggfortify)**

```
autoplot(model.with.age)
```

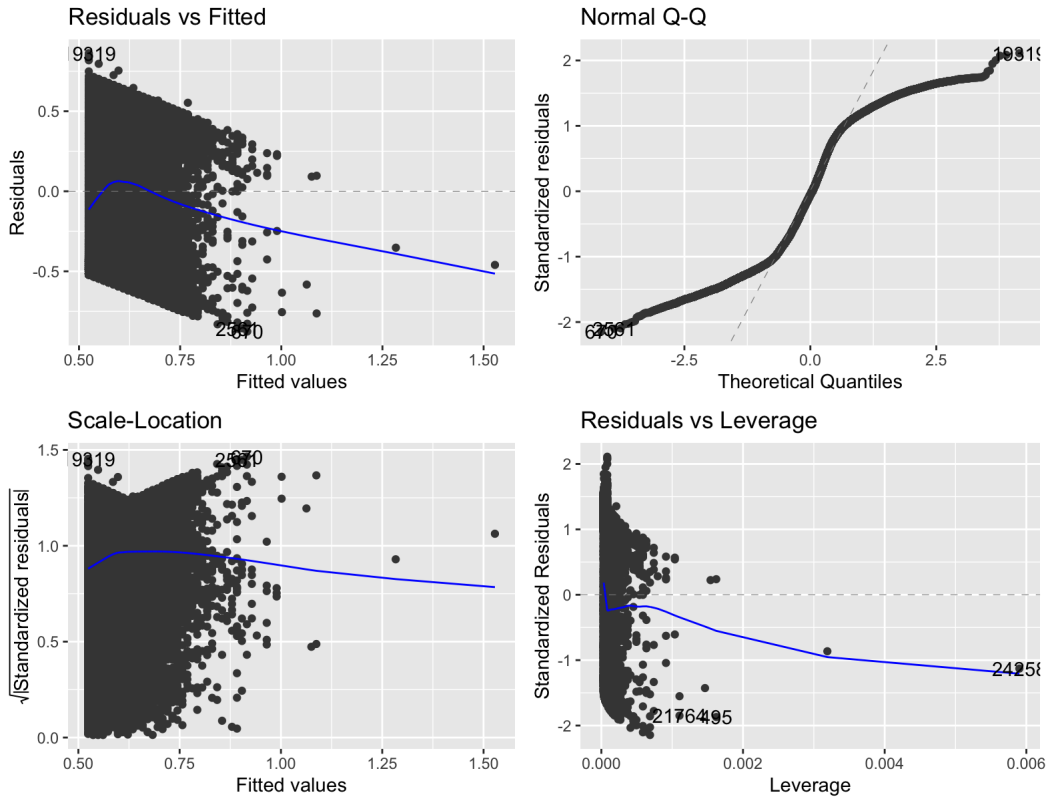

Figure 31: Check conditions for linear regression

For a linear regression model to produce valid results, these conditions must be met. For checking the linearity of the relationship between the independent and dependent variables, residuals vs fitted plots are used, normal Q-Q plots are used for checking residual normality, scale-location plots are used to check homoscedasticity, and residuals vs leverage plots are used to identify influential data points that may impact regression results significantly. The results of linear regression may not be valid if these assumptions are not met, and the model may not accurately represent the data.

Method 2: Manually

1. Linearity

```
plot(model.with.age,1)
```

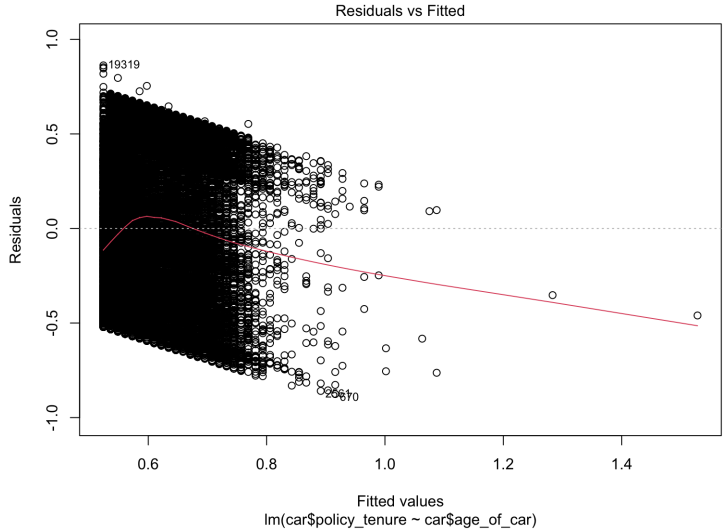

Figure 32: Check conditions for model policy_tenture age of car - Linearity

According to the figure 32, it's not linear.

2. Nearly Normal Residuals

```
plot(model.with.age,2)
```

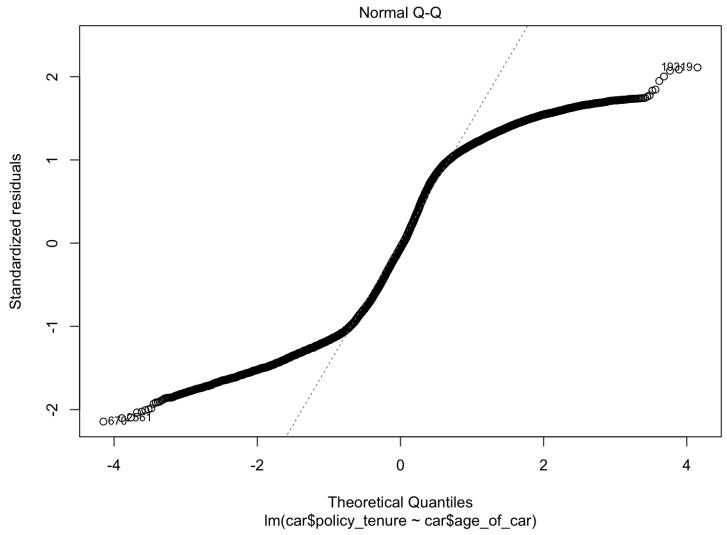

Figure 33: Check conditions for model policy_tenture age of car - Nearly Normal Residuals

According to the figure 33, it's short tail so, not normal.

3. Constant Variability

```
1 plot(model.with.age,3)
```

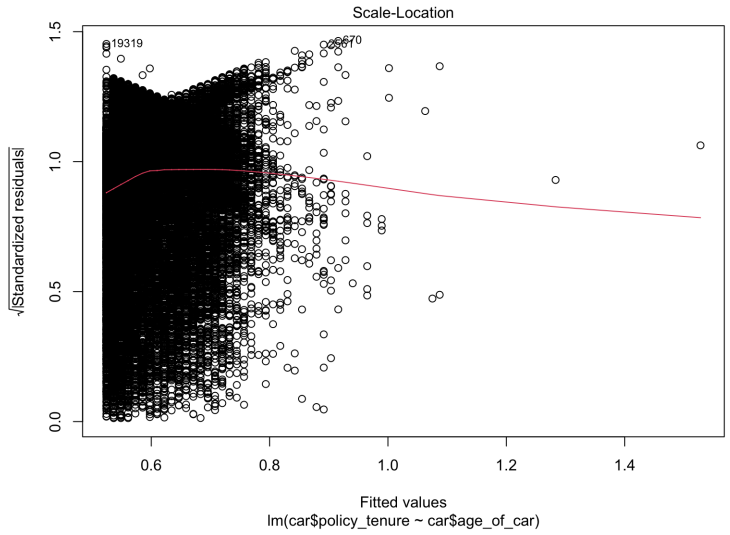

Figure 34: Check conditions for model policy_tenture age of car - Constant Variability

According to the figure 34, may could say that, it's constant variability.

We continue despite the conditions not being met.

4.2.2 b

Compute the least squares regression.

```r
model.with.age <- lm(car$policy_tenure ~ car$age_of_car)
```

```
> summary(model.with.age)

Call:
lm(formula = car$policy_tenure ~ car$age_of_car)

Residuals:
     Min       1Q   Median       3Q      Max
-0.87577 -0.39872 -0.02297  0.40773  0.86233

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.524187   0.003736  140.30   <2e-16 ***
car$age_of_car 1.224236   0.041699   29.36   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4085 on 29998 degrees of freedom
Multiple R-squared:  0.02793,   Adjusted R-squared:  0.0279
F-statistic: 861.9 on 1 and 29998 DF,  p-value: < 2.2e-16
```

Figure 35: summary of model.with.age

4.2.3 c

Write the predictive equation for the response variable and interpret its parameters.

$$\bar{policy\_tenure} = 0.524187 + 1.224236 \times age\_of\_car$$

As a result of the normalization of our dataset, we do not have a unit for the car's age or the duration of its policy.
Interpretation:
1. Intercept: The tenure of a policy for a new, recently produced car is approximately 0.524187 (years).
2. Slope: On average, the policy tenure increases by 1.224236 units with each unit increase in car age.

4.2.4 d

Draw a scatter plot of the relation between these two variables overlaid with this least-squares fit as a dashed line.

Figure 36: scatter plot of the relation between response and explanatory variables.

2. displacement

4.2.5 a

Check the Linearity, Nearly Normal Residuals, and Constant Variability conditions in R

Method 1: use **library(ggfortify)**

```
1 autoplot(model.with.dis)
```

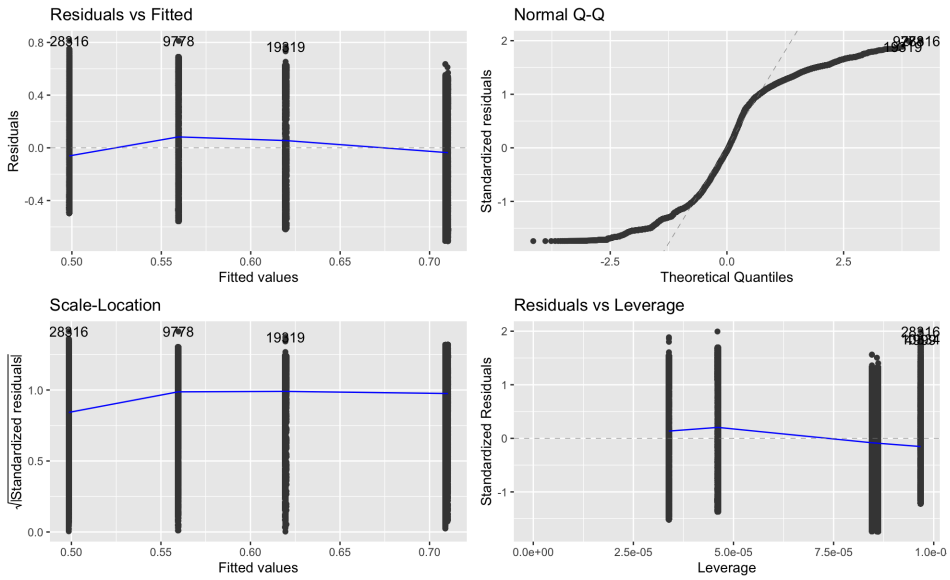

Figure 37: Check conditions for linear regression

Method 2: Manually

1. Linearity

```
plot(model.with.dis,1)
```

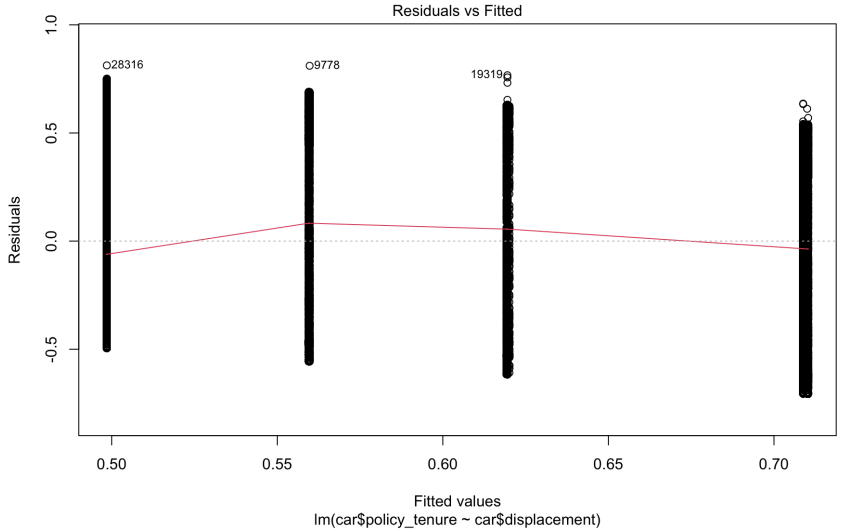

Figure 38: Check conditions for model policy_tenture displacement - Linearity

According to the figure 38, it's not linear.

2. Nearly Normal Residuals

```
plot(model.with.dis,2)
```

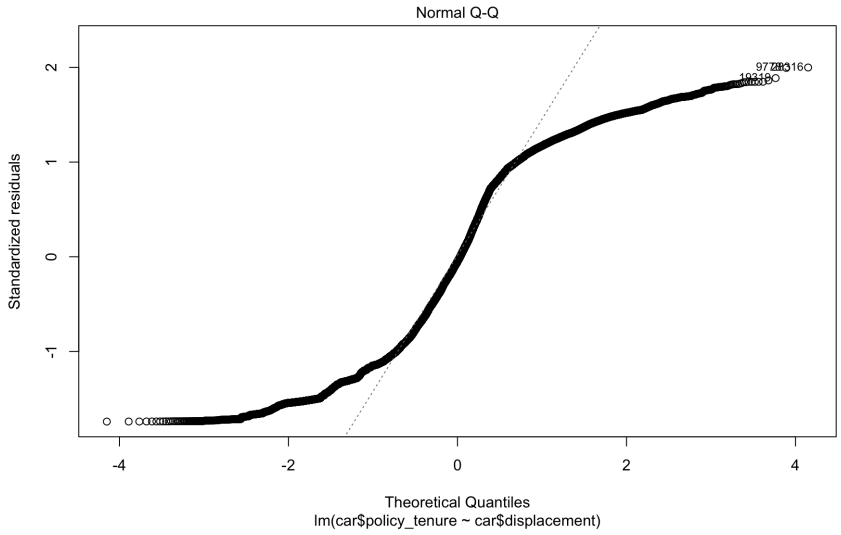

Figure 39: Check conditions for model policy_tenture displacement - Nearly Normal Residuals

According to the figure 39, it's short tail so, not normal.

3. Constant Variability

```
1 plot(model.with.dis,3)
```

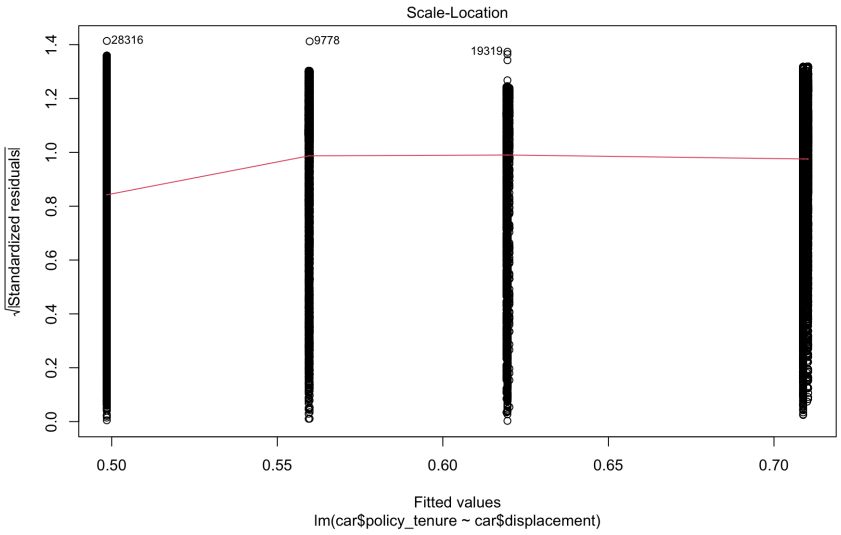

Figure 40: Check conditions for model policy_tenture displacement - Constant Variability

According to the figure 58, may could say that, it's constant variability.

We continue despite the conditions not being met.

4.2.6 b

Compute the least squares regression.

```r
model.with.age <- lm(car$policy_tenure ~ car$age_of_car)
```

```
> summary(model.with.dis)

Call:
lm(formula = car$policy_tenure ~ car$displacement)

Residuals:
     Min       1Q   Median       3Q      Max
-0.70753 -0.38774 -0.02114  0.40083  0.81250

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      2.584e-01  1.052e-02   24.56   <2e-16 ***
car$displacement 3.017e-04  8.819e-06   34.21   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4065 on 29998 degrees of freedom
Multiple R-squared:  0.03755,   Adjusted R-squared:  0.03752
F-statistic:  1170 on 1 and 29998 DF,  p-value: < 2.2e-16
```

Figure 41: summary of model.with.dis

4.2.7 c

Write the predictive equation for the response variable and interpret its parameters.

$$\bar{policy\_tenure} = 0.2584 + 0.0003017 \times displacement$$

As a result of the normalization of our dataset, we do not have a unit for policy_tenure
Interpretation:
1. Intercept: Policy tenure for a car with zero displacement, approximately 0.2584 (years).
2. Slope: On average, the policy tenure increases by 0.0003017 units with each unit increase in displacement.

4.2.8 d

Draw a scatter plot of the relation between these two variables overlaid with this least-squares fit as a dashed line.

Figure 42: scatter plot of the relation between response and explanatory variables.

4.3 C

By using the previous part results, try to explain which explanatory variable is the more significant predictor

In general, the more significant predictor is the one with the lowest p-value and highest *Adjusted* R^2.

Since variable displacement has a much higher *Adjusted* R^2, it explains policy tenure variability much better than car age. Additionally, both p-values are significant (there is no difference in p-value) and close to zero. Displacement appears to be more significant than the age of the car.

Choosing one variable, we choose displacement, but both don't meet conditions and have low *Adjusted* R^2.

4.4 D

Now, Compare your models, once using adjusted R2 and another time by ANOVA table. Explain results.

Adjusted R^2

	Adjusted R^2	p-value
age of car	0.0279	2.2e-16
displacement	0.03752	2.2e-16

According to *Adjusted* R^2, displacement explains 0.03752 of the model's variability (which is quite low).

According to *Adjusted* R^2, the age of the car explains 0.0279 of the model's variability (which is extremely low).

Based on the summary of the two models, the model with displacement has a much higher *Adjusted* R^2 and therefore better explains policy tenure variability.

ANOVAtable

```
> anova(model.with.age)
Analysis of Variance Table

Response: car$policy_tenure
                Df Sum Sq Mean Sq F value    Pr(>F)
car$age_of_car   1  143.9 143.866  861.93 < 2.2e-16 ***
Residuals    29998 5007.0   0.167
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> anova(model.with.dis)
Analysis of Variance Table

Response: car$policy_tenure
                  Df Sum Sq Mean Sq F value    Pr(>F)
car$displacement   1  193.4 193.433  1170.5 < 2.2e-16 ***
Residuals      29998 4957.4   0.165
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figure 43: ANOVA for age of car and displacement

According to the two anova tables, both models have significant p-values close to zero

Using the p-value, we can reject the null hypothesis that suggests there is no relationship between policy tenure and displacement.
Using the p-value, we can reject the null hypothesis that suggests there is no relationship between policy tenure and displacement.

Calculating R^2 to compare two models gives the same result as *Adjusted* R^2 since n-1 and n-k-1 are almost equal.

4.5 E

According to the results that you found in the previous parts, list the features of a good predictor.

There are several methods for selecting the most appropriate predictors to use in a regression model. There are strengths and weaknesses to each method, and the choice of which methods to use depends on the context.

Valid predictors should have the following features:

In order to be linear regression, it must meet the following conditions:
1. Linearly related to the response variable.
2. There should be a nearly normal distribution of residuals.
3. There should be relatively constant variability around the regression line

The p-value should be significant, in other words, it should explain well how the response variable varies.

The adjusted R2 measures how well the model fits the data. In addition, it tends to choose too many predictor variables, which may not be suitable for forecasting.

Using cross-validation, data is divided into training and validation sets, and the validation set is used to estimate the model's performance. Models with the smallest mean squared error (MSE) are most suitable.

4.6 F

Choose a random sample of 100 data points from the dataset.

4.6.1 a

By 90 percent of data, Build two Linear Regression models and design hypothesis tests to see if these explanatory variables are a significant predictor of the response variable or not.

Setting the hypothesis:

H_0 :*The explanatory variable is not a significant predictor of the response variable* $\beta = 0$
H_a :*The explanatory variable is a significant predictor of the response variable*, $\beta \neq 0$

After taking a sample of size 100, we build two models.

```
set.seed(1)
car_sample <- sample_n(car, 100)

train.car <- car_sample[1:90, ]
test.car <- car_sample[91:100, ]

model2.with.age<- lm(policy_tenure ~ age_of_car, train.car)
model2.with.dis<- lm(policy_tenure ~ displacement, train.car)
```

```
> summary(model2.with.age)

Call:
lm(formula = policy_tenure ~ age_of_car, data = train.car)

Residuals:
     Min       1Q   Median       3Q      Max
-0.68165 -0.39310 -0.04732  0.42517  0.70795

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.46305    0.07158   6.469 5.37e-09 ***
age_of_car   1.97412    0.86146   2.292   0.0243 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4297 on 88 degrees of freedom
Multiple R-squared:  0.05632,   Adjusted R-squared:  0.04559
F-statistic: 5.251 on 1 and 88 DF,  p-value: 0.02432
```

Figure 44: summary of model2.with.age

```
> summary(model2.with.dis)

Call:
lm(formula = policy_tenure ~ displacement, data = train.car)

Residuals:
     Min       1Q   Median       3Q      Max
-0.67103 -0.41548 -0.02897  0.45692  0.66121

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.3350170  0.1933870   1.732   0.0867 .
displacement 0.0002269  0.0001671   1.358   0.1779
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4378 on 88 degrees of freedom
Multiple R-squared:  0.02053,   Adjusted R-squared:  0.009395
F-statistic: 1.844 on 1 and 88 DF,  p-value: 0.1779
```

Figure 45: summary of model2.with.dis

According to the summary above, both explanatory variables are significant predictors of the response.

4.6.2 b

Calculate the 95% confidence interval for the slope of the relationship between response variable and explanatoryvariables. Interpret these CIs.

```
b1 <- model2.with.age$coefficients[2]
se1 <- summary(model2.with.age)$coefficient[4]
tstar <- abs(qt(0.025, df = 90-1-1))
me1 <- se1 * tstar

lowerinterval1 <- b1 - me1
lowerinterval1

upperinterval1 <- b1 + me1
upperinterval1


b2 <- model2.with.dis$coefficients[2]
se2 <- summary(model2.with.dis)$coefficient[4]
tstar <- abs(qt(0.025, df = 90-1-1))
me2 <- se2 * tstar

lowerinterval2 <- b2 - me2
lowerinterval2

```

```
upperinterval2 <- b2 + me2
upperinterval2
```

As a result, we are 95% confident that for each 1 unit increase in age of car, the response variable (policy tenure) increases by 0.2621554 to 3.686083 (years).

As a result, we are 95% confident that for each 1 unit increase in displacement, the response variable (policy tenure) increases by -0.0001051642 to 0.0005590302 (years).

4.6.3 c

Use your models to predict the values of the response variable for the remaining percent of samples.

Predictions of test data are based on two models.

```
predict1 <- predict(model2.with.age,select(test.car,age_of_car))
predict2 <- predict(model2.with.dis,select(test.car,displacement))
```

```
> predict1
       91        92        93        94        95        96        97        98        99       100
0.4827864 0.6604571 0.4827864 0.5617511 0.6801982 0.5617511 0.6209747 0.7394218 0.8578690 0.4827864
> predict2
       91        92        93        94        95        96        97        98        99       100
0.5156557 0.6066558 0.5156557 0.5156557 0.6066558 0.5156557 0.6066558 0.5617231 0.6066558 0.5156557
```

Figure 46: Two models' predictions

4.6.4 d

Compare the predicted values with actuals. Report success rate.

```
actual1  <- select(test.car,policy_tenure)
diff1    <- round(abs(predict1-actual1),1)
success.rate1 <- length(diff1[diff1== 0])/nrow(diff1)

actual2  <- select(test.car,policy_tenure)
diff2    <- round(abs(predict2-actual2),1)
success.rate2 <- length(diff2[diff2== 0])/nrow(diff2)
```

success rate for model 1 (age of the car) is: 10%
success rate for model 2 (displacement) is: 0%

Since the sample was random, model 1 performed better here than in previous parts. This sample's success rate suggests that the car's age is a better predictor.

5 Question 5

Consider the response variable you selected in the previous question. You can use as many explanatory variables as you deem necessary.

5.1 A

Plot a correlogram for explanatory variables and discuss the correlation between them. Could you find which explanatory variable plays a more significant role in prediction

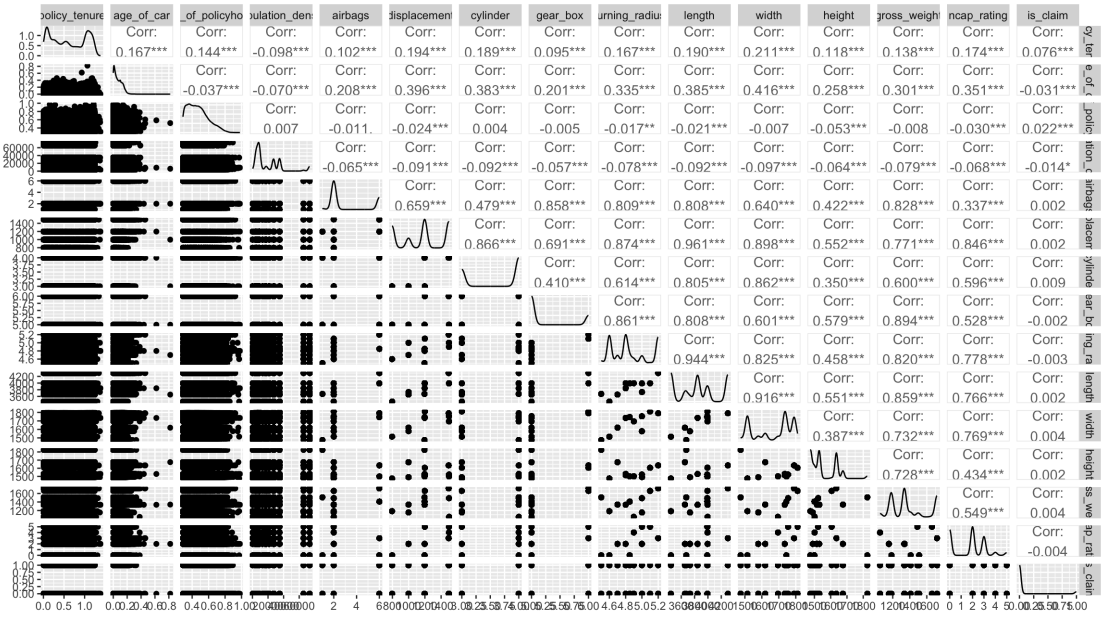

Figure 47: Correlogram

Since this dataset is collected for a binary response variable, all explanatory variables have low correlations with our response variable. The response variable in this question must be a continuous numerical variable, so I chose policy tenure. With other variables in this dataset, we cannot predict this variable so well.

After selection, width has the highest correlation with policy tenure, and length and displacement have the highest correlation with width, so one of these variables remains in the last model.

After that, the variables age_of_car, cylinder and NCAP_rate have the highest correlation with policy tenure.

Choose policy_tenure as the response variable and age_of_car, NCAP_rate, displacement, width and height as explanatory variables.

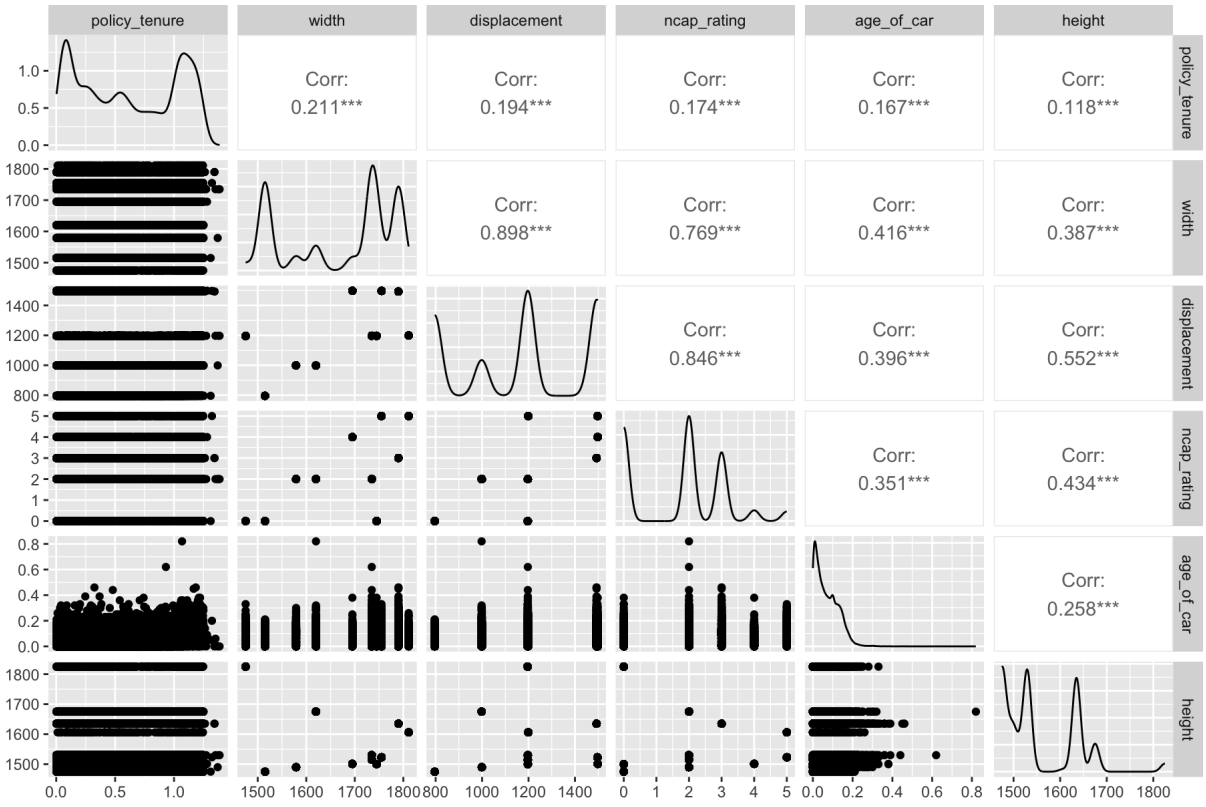

Figure 48: Correlogram

5.2 B

Develop a multiple linear regression model for the response variable using explanatory variables you found in part A

```r
car.sample <- data %>% select(policy_tenure,width, displacement, ncap_
    rating,age_of_car, height)
model <- lm(policy_tenure ~ . , data = car.sample)
summary(model)
```

```
> summary(model)

Call:
lm(formula = policy_tenure ~ ., data = car.sample)

Residuals:
     Min       1Q   Median       3Q      Max
-0.85750 -0.37779 -0.01878  0.39404  0.86881

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -8.326e-01  9.220e-02  -9.030  < 2e-16 ***
width         6.863e-04  5.032e-05  13.638  < 2e-16 ***
displacement -8.898e-05  2.699e-05  -3.297  0.00098 ***
ncap_rating   8.898e-03  3.154e-03   2.821  0.00480 **
age_of_car    6.702e-01  4.558e-02  14.703  < 2e-16 ***
height        2.159e-04  3.698e-05   5.838 5.34e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4032 on 29994 degrees of freedom
Multiple R-squared:  0.05346,   Adjusted R-squared:  0.0533
F-statistic: 338.8 on 5 and 29994 DF,  p-value: < 2.2e-16
```

Figure 49: Summary of multiple linear regression model

$$\begin{aligned} policy\_tenure = -0.8326 &+ 0.0006863 \times width \\ &- 0.00008898 \times displacement \\ &+ 0.008898 \times ncap\_rate \\ &+ 0.6702 \times age\_of\_car \\ &+ 0.0002159 \times height \end{aligned}$$

5.3 C

How well do you think your model fits the data?

According to the summary of the model, the p-value of the F-statistic is almost zero and 5.3% of the variability in the response variable is explained by this model. Out of the 20 variables, five were used to explain nearly 5.3% of the variability in our model.

The model could not be fitted to the data.

5.4 D

Develop the "best" possible multiple linear regression model for the response variable using different approaches and metrics

In stepwise model selection, "backward" and "forward" refer to the direction in which the algorithm moves through the variables in the model.

Backward Selection: The backward selection method starts with all the variables in the model and removes the variable with the highest p-value (least significant) at each step, until all remaining variables have a p-value below a certain threshold.

Forward Selection: The forward selection method starts with an empty model and adds the variable with the lowest p-value (most significant) at each step, until all variables have been considered or the addition of new variables does not improve the model fit.

Both methods are used to select a subset of variables from a larger set to improve model fit and prevent overfitting. Using the selected independent variables, we need to find the best adjusted R-squared, which is a statistical measure of how well the model predicts the outcome.

Backward elimination:
1. Perform backward elimination with *Adjusted* R^2: (follow method 1 in code)

```
> summary(best.model.back)

Call:
lm(formula = build_model("policy_tenure", variables), data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.90207 -0.35822 -0.00267  0.36744  0.91295

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)         -3.842e+00  2.869e-01 -13.393  < 2e-16 ***
age_of_car           5.878e-01  4.548e-02  12.923  < 2e-16 ***
age_of_policyholder  5.142e-01  1.857e-02  27.687  < 2e-16 ***
population_density  -1.706e-06  1.297e-07 -13.148  < 2e-16 ***
airbags             -1.496e-02  4.637e-03  -3.226 0.001255 **
displacement         5.016e-05  7.197e-05   0.697 0.485845
gear_box            -7.565e-02  2.045e-02  -3.699 0.000217 ***
turning_radius       5.619e-01  7.160e-02   7.847 4.40e-15 ***
length              -1.757e-04  1.108e-04  -1.585 0.112946
width                1.001e-03  1.051e-04   9.524  < 2e-16 ***
height               9.065e-04  6.928e-05  13.083  < 2e-16 ***
gross_weight        -3.459e-04  4.566e-05  -7.575 3.70e-14 ***
ncap_rating         -4.256e-02  7.760e-03  -5.485 4.18e-08 ***
is_claim             1.256e-01  9.327e-03  13.468  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3943 on 29986 degrees of freedom
Multiple R-squared:  0.09511,   Adjusted R-squared:  0.09472
F-statistic: 242.4 on 13 and 29986 DF,  p-value: < 2.2e-16
```

Figure 50: Summary of multiple linear regression model - backward elimination use Adjusted R-squared

2. Perform backward elimination with p-value:(follow method 3 in code)

```
> summary(best.model.back.pvalue)

Call:
lm(formula = build_model("policy_tenure", variables), data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.90216 -0.35823 -0.00281  0.36741  0.91313

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         -3.778e+00  2.721e-01 -13.887  < 2e-16 ***
age_of_car           5.880e-01  4.548e-02  12.927  < 2e-16 ***
age_of_policyholder  5.147e-01  1.856e-02  27.734  < 2e-16 ***
population_density  -1.707e-06  1.297e-07 -13.158  < 2e-16 ***
airbags             -1.508e-02  4.634e-03  -3.254  0.00114 **
gear_box            -8.056e-02  1.920e-02  -4.196 2.72e-05 ***
turning_radius       5.282e-01  5.293e-02   9.981  < 2e-16 ***
length              -1.056e-04  4.666e-05  -2.264  0.02359 *
width                9.558e-04  8.269e-05  11.558  < 2e-16 ***
height               8.881e-04  6.408e-05  13.860  < 2e-16 ***
gross_weight        -3.375e-04  4.405e-05  -7.661 1.89e-14 ***
ncap_rating         -3.875e-02  5.510e-03  -7.033 2.07e-12 ***
is_claim             1.256e-01  9.326e-03  13.470  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3943 on 29987 degrees of freedom
Multiple R-squared:  0.0951,    Adjusted R-squared:  0.09473
F-statistic: 262.6 on 12 and 29987 DF,  p-value: < 2.2e-16
```

Figure 51: Summary of multiple linear regression model - backward elimination use p-value

Forward selection:
1. Perform forward selection with *Adjusted* R^2: (follow method 2 in code)

```
> summary(best.model.for)

Call:
lm(formula = build_model("policy_tenure", best.variable), data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.90139 -0.35804 -0.00314  0.36752  0.91320

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         -3.929e+00  3.113e-01 -12.621  < 2e-16 ***
width                1.187e-03  2.773e-04   4.279 1.88e-05 ***
age_of_policyholder  5.144e-01  1.857e-02  27.694  < 2e-16 ***
age_of_car           5.865e-01  4.552e-02  12.887  < 2e-16 ***
population_density  -1.707e-06  1.297e-07 -13.157  < 2e-16 ***
is_claim             1.257e-01  9.327e-03  13.474  < 2e-16 ***
height               8.166e-04  1.422e-04   5.745 9.30e-09 ***
gross_weight        -3.076e-04  6.992e-05  -4.399 1.09e-05 ***
turning_radius       5.634e-01  7.163e-02   7.865 3.81e-15 ***
ncap_rating         -6.997e-02  3.864e-02  -1.811 0.070194 .
airbags             -2.925e-02  2.027e-02  -1.443 0.149102
gear_box            -7.310e-02  2.075e-02  -3.523 0.000427 ***
length              -1.916e-04  1.130e-04  -1.695 0.090016 .
displacement         2.969e-04  3.484e-04   0.852 0.394023
cylinder            -8.003e-02  1.105e-01  -0.724 0.469078
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3943 on 29985 degrees of freedom
Multiple R-squared:  0.09513,   Adjusted R-squared:  0.0947
F-statistic: 225.2 on 14 and 29985 DF,  p-value: < 2.2e-16
```

Figure 52: Summary of multiple linear regression model - forward elimination use Adjusted R-squared

2. Perform forward elimination with p-value: (follow method 4 in code)

```
> summary(best.model.for.pvalue)

Call:
lm(formula = policy_tenure ~ age_of_car + age_of_policyholder +
    population_density + airbags + displacement + cylinder +
    gear_box + turning_radius + length + width + height + gross_weight +
    ncap_rating + is_claim, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.90139 -0.35804 -0.00314  0.36752  0.91320

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)         -3.929e+00  3.113e-01 -12.621  < 2e-16 ***
age_of_car           5.865e-01  4.552e-02  12.887  < 2e-16 ***
age_of_policyholder  5.144e-01  1.857e-02  27.694  < 2e-16 ***
population_density  -1.707e-06  1.297e-07 -13.157  < 2e-16 ***
airbags             -2.925e-02  2.027e-02  -1.443 0.149102
displacement         2.969e-04  3.484e-04   0.852 0.394023
cylinder            -8.003e-02  1.105e-01  -0.724 0.469078
gear_box            -7.310e-02  2.075e-02  -3.523 0.000427 ***
turning_radius       5.634e-01  7.163e-02   7.865 3.81e-15 ***
length              -1.916e-04  1.130e-04  -1.695 0.090016 .
width                1.187e-03  2.773e-04   4.279 1.88e-05 ***
height               8.166e-04  1.422e-04   5.745 9.30e-09 ***
gross_weight        -3.076e-04  6.992e-05  -4.399 1.09e-05 ***
ncap_rating         -6.997e-02  3.864e-02  -1.811 0.070194 .
is_claim             1.257e-01  9.327e-03  13.474  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3943 on 29985 degrees of freedom
Multiple R-squared:  0.09513,   Adjusted R-squared:  0.0947
F-statistic: 225.2 on 14 and 29985 DF,  p-value: < 2.2e-16
```

Figure 53: Summary of multiple linear regression model - forward elimination use p-value

To select appropriate predictors, I use four methods, and the most effective module is ‘best.model.back.pvalue‘, which includes the following predictors:

```
Coefficients:
       (Intercept)          age_of_car  age_of_policyholder   population_density             airbags
         -3.778e+00           5.880e-01           5.147e-01          -1.707e-06          -1.508e-02
          gear_box       turning_radius              length               width              height
         -8.056e-02           5.282e-01          -1.056e-04           9.558e-04           8.881e-04
       gross_weight         ncap_rating            is_claim
         -3.375e-04          -3.875e-02           1.256e-01
```

Figure 54: best predictors

5.5 E

Use 5-fold cross-validation and compare the model's RMSE (part B and D). How do you interpret these values?

```
library(caret)
train_control <- trainControl(method = "cv", number = 5)

# part B
predictors <- names(model_partB$coefficients)
predictors <- predictors[!predictors %in% '(Intercept)']

model.last <- train(build_model('policy_tenure', predictors), data =
    data , method = "lm",trControl = train_control)
model.last


# part D
predictors <- names(best.model.back.pvalue$coefficients)
predictors <- predictors[!predictors %in% '(Intercept)']

model.last2 <- train(build_model('policy_tenure', predictors), data =
    data , method = "lm",trControl = train_control)
model.last2
```

```
> model.last <- train(build_model('policy_tenure', predictors), data = data , method = "lm",trControl = train_control)
> model.last
Linear Regression

30000 samples
    5 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 24000, 24000, 24000, 24000, 24000
Resampling results:

  RMSE       Rsquared    MAE
  0.4031737  0.05338487  0.361134

Tuning parameter 'intercept' was held constant at a value of TRUE
```

Figure 55: model in part B

```
> model.last2 <- train(build_model('policy_tenure', predictors), data = data , method = "lm",trControl = train_control)
> model.last2
Linear Regression

30000 samples
   12 predictor

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 24000, 24000, 24000, 24000, 24000
Resampling results:

  RMSE       Rsquared    MAE
  0.3943709  0.09421365  0.3500927

Tuning parameter 'intercept' was held constant at a value of TRUE
```

Figure 56: model in part D - best model

Due to the low RMSE in 'part D', we can say we're doing a good job and the backward method produces a model with a smaller RMSE, which indicates that it has less error.

5.6 F

Check diagnostics for your model in part D (Three conditions: 1. Linearity, 2. Nearly normal residuals, and 3. Constant variability) and explain if this is a reliable model or not

Conditions for linear regression :

1. Linear relationship: There exists a linear relationship between the independent variable, x, and the dependent variable, y.

2. Independence: The residuals are independent. In particular, there is no correlation between consecutive residuals in time series data.

3. Homoscedasticity: The residuals have constant variance at every level of x.

4. Normality: The residuals of the model are normally distributed.

Method 1: use **library(ggfortify)**

```
autoplot(best.model.back.pvalue)
```

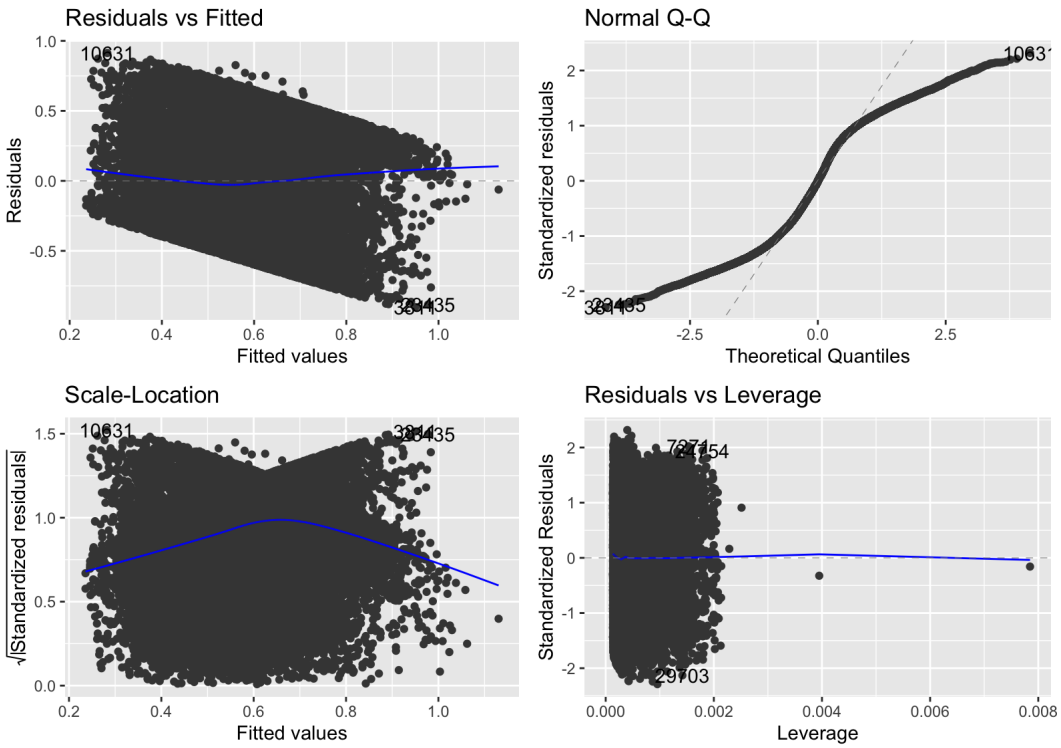

Figure 57: Check conditions for linear regression

For a linear regression model to produce valid results, these conditions must be met. For checking the linearity of the relationship between the independent and dependent variables, residuals vs fitted plots are used, normal Q-Q plots are used for checking residual normality, scale-location plots are used to check homoscedasticity, and residuals vs leverage plots are used to identify influential data points that may impact regression results significantly. The results of linear regression may not be valid if these assumptions are not met, and the model may not accurately represent the data.

Method 2: Manually

1. Linearity

```
plot(best.model.back.pvalue,1)
```

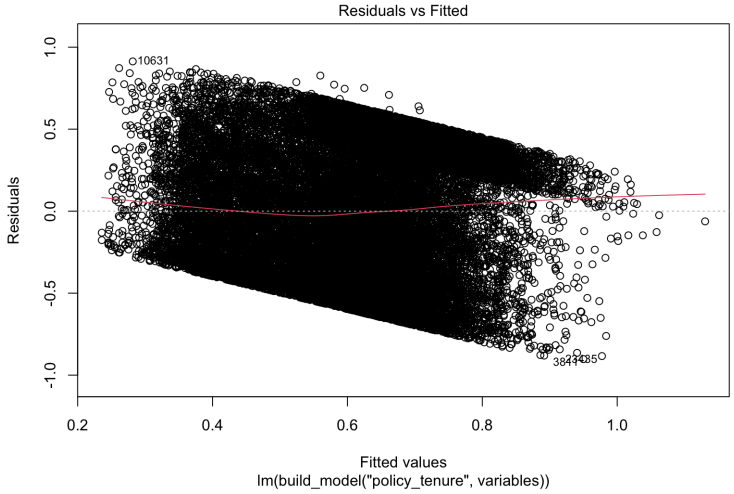

Figure 58: Check conditions for model policy_tenture - Linearity

According to the figure 58, it's not linear.

2. Nearly Normal Residuals

```
plot(best.model.back.pvalue,2)
```

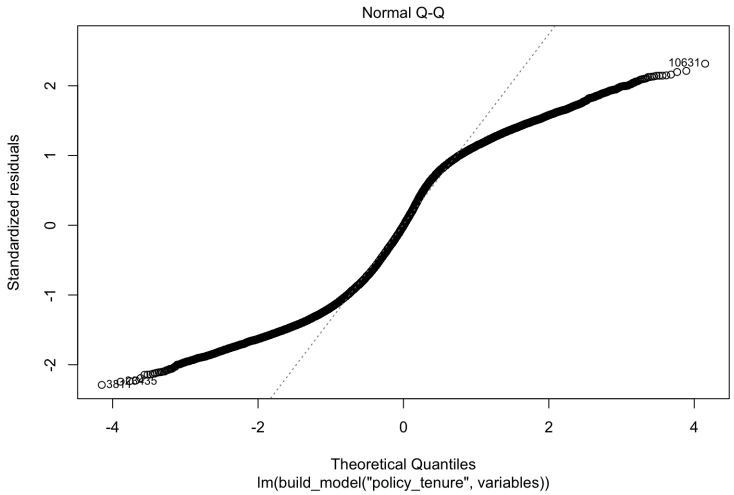

Figure 59: Check conditions for model policy_tenture - Nearly Normal Residuals

According to the figure 59, it's short tail so, not normal.

3. Constant Variability

```
plot(best.model.back.pvalue,3)
```

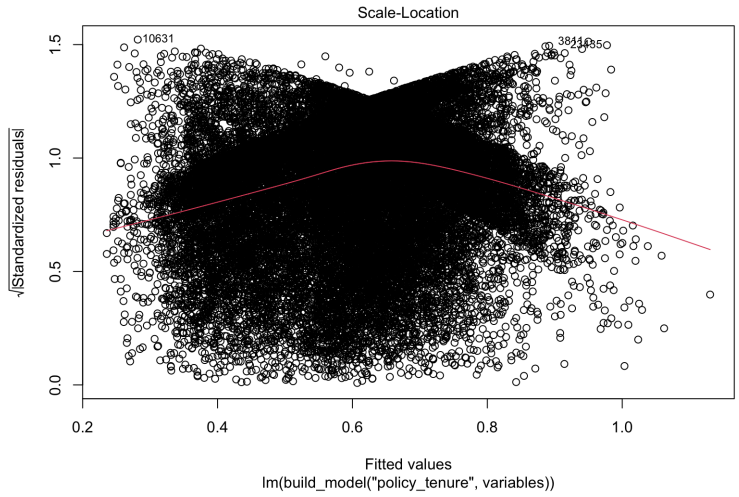

Figure 60: Check conditions for model policy_tenture - Constant Variability

According to the figure 60, may could say that, it's constant variability.

Conditions aren't met.
It is not a reliable model.

5.7 G

What percent of the variation in the response variable is explained by the model (part B and D)?

In part B I use 5 number of variables and $AdjustedR^2 = 0.0533$ which means only 5.3% of the variation in the response variable is explained by this model.

In part D I use 13 number of variables and $AdjustedR^2 = 0.09473$ which means only 9.4% of the variation in the response variable is explained by this model.

6 R codes

R Codes

```r
# Import Libraries

library(tidyverse)
library(stats)
theme_set(theme_minimal())


# Load the dataset
car <- read.csv("./Downloads/UT/Statistical Inference/Project/Phase II/Car_Insurance_Claim_Prediction.csv")

#-----> Question 1 <-----
# categorical variables
car.categorical <- car %>% select_if(negate(is.numeric))
colnames(car.categorical)

# number of unique variables for each columns
car.categorical.t <- as.data.frame(t(car.categorical))
apply(car.categorical.t, 1, function(car.categorical) length(unique(car.categorical)))

# numerical variables
car.numeric <- car %>% select_if((is.numeric))
colnames(car.numeric)

# number of unique variables for each columns
car.numeric.t <- as.data.frame(t(car.numeric))
apply(car.numeric.t, 1, function(car.numeric) length(unique(car.numeric)))

# convert some types of numberical variables
car$population_density <- as.factor(car$population_density)
car$airbags <- as.factor(car$airbags)
car$displacement <- as.factor(car$displacement)
car$cylinder <- as.factor(car$cylinder)
car$turning_radius <- as.factor(car$turning_radius)
car$length <- as.factor(car$length)
car$width <- as.factor(car$width)
car$height <- as.factor(car$height)
car$gross_weight <- as.factor(car$gross_weight)
car$ncap_rating <- as.factor(car$ncap_rating)

# categorical variables
car.categorical <- car %>% select_if(negate(is.numeric))
colnames(car.categorical)

# number of unique variables for each columns
car.categorical.t <- as.data.frame(t(car.categorical))
apply(car.categorical.t, 1, function(car.categorical) length(unique(car.categorical)))


```

```r
# sampling
car_sample <- sample_n(car.categorical, 1000)

# calc each two combinations of categorical variables
n <- 2:21
for(i in n){
  x <- i+1
  m <- x:22
  for(j in m){
    print("#------")
    print(paste0(colnames(car_sample)[i],", ",colnames(car_sample)[j])
   )
    print(table(car_sample[,i],car_sample[,j]))
    print("------#")
  }
}

# change categorical to numerical
car_sample$ncap_rating <- sapply(car_sample$ncap_rating, unclass)

# merge 3 last columns
car_sample <- within(car_sample, {
  ncap_rating.new <- NA # need to initialize variable
  ncap_rating.new[ncap_rating == 1] <- "0"
  ncap_rating.new[ncap_rating == 2] <- "2"
  ncap_rating.new[ncap_rating >= 3] <- "more than 2"

} )

# show the new contingency table
print(paste0(colnames(car_sample)[10],", ",colnames(car_sample)[23]))
print(table(car_sample[,10],car_sample[,23]))

# select the two categorical variables
ncap_rating <- car_sample$ncap_rating.new
is_parking_camera <- car_sample$is_parking_camera

unique(ncap_rating)
unique(is_parking_camera)

#Part A ---------
# contingency table of ncap_rating and is_parking_camera
contingency_table <- addmargins(table(is_parking_camera, ncap_rating))
contingency_table

# calculate proportions
contingency_table_p <- contingency_table
contingency_table_p[1,] <- round((contingency_table[1,]/contingency_
   table[1,4]),4)
contingency_table_p[2,] <- round((contingency_table[2,]/contingency_
   table[2,4]),4)
contingency_table_p[1:2,1:3]

#calculate confidence intervar for each ratings
n1 <- contingency_table[1,4]
n2 <- contingency_table[2,4]

```

```r
for(i in 1:length(colnames(contingency_table_p[1:2,1:3]))){

  p1 <- contingency_table_p[1,i]
  p2 <- contingency_table_p[2,i]

  pointest <- p1-p2

  se <- sqrt(((p1*(1-p1))/n1)+((p2*(1-p2))/n2))

  c <- 0.95
  zscore <- qnorm((1-c)/2,lower.tail = FALSE)

  me <- zscore * se

  lowerinterval <- round(pointest - me,3)
  upperinterval <- round(pointest + me,3)

  print(paste0("Cofidence Interval for ", colnames(contingency_table_p
   )[i]," start(s) : (",lowerinterval, ", ", upperinterval, ")"))
}


#Part B ---------

# Perform a chi-squared test of independence on the contingency table
chi_squared_test <- chisq.test(contingency_table)

# Print the p-value of the test to determine if the two variables are
   independent
print(chi_squared_test$p.value)


#-----> Question 2 <-----
# chose transmission_type as binary variable

car_sample <- sample_n(car, 15)

transmission <- car_sample$transmission_type

Manual <- transmission[transmission =='Manual']
Automatic <- transmission[transmission =='Automatic']

frequencies<-c(length(Manual),length(Automatic))
percentage  <- round(100*frequencies/sum(frequencies), 2)

transmission.categorized <- data.frame(types = c("Manual", "Automatic"
   ),percentage = percentage, value = frequencies)

ggplot(data=transmission.categorized, aes(x=types, y=percentage, fill=
   types)) +
  geom_bar(stat="identity", alpha = 0.7, width = 0.6)+
  labs(title="Barplot of Transmission Type")+
  geom_text(aes(label=paste(percentage, "%")),vjust=-0.4,size=4)

# run simulation
# method 1
p.hat <- table(transmission)[2]/15
```

```r
transmission.simulation <-  data.frame(replicate(n = 1000, mean(sample
    (levels(as.factor(transmission)), size = 15, replace = TRUE)=='
    Manual')))
p_value <- mean(transmission.simulation >= p.hat)
p_value

# method 2
source("./Downloads/UT/Statistical Inference/Project/Phase II/
    inference.R")
inference(transmission,est="proportion",
          type="ht", success = "Manual",
          method = "simulation",
          null=0.5,
          alternative = "greater")


#-----> Question 3 <-----
# Part A ---------

# random sampling
sample1 <- sample_n(car, 100)$ncap_rating
Number <- table(sample1)

# calculate proportions randomly sample
Proportion <- round((prop.table(Number)),3)
car.sample.unbiased <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.unbiased


# biased sampling
pro <- ifelse(car$ncap_rating > 3 , 0.9, 0.1)
sample2 <- sample(car$ncap_rating, 100, prob = pro)
Number <- table(sample2)

# calculate proportions
Proportion <- round((prop.table(Number)),3)
car.sample.biased <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.biased

# merge 2 last columns
sample1.new <- ifelse(sample1 > 3, "more than 3", sample1)
sample2.new <- ifelse(sample2 > 3, "more than 3", sample2)

# sample1 : random sampling
Number <- table(sample1.new)
Proportion <- round((prop.table(Number)),3)
car.sample.unbiased.new <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.unbiased.new

# sample2 : biased sampling
Number <- table(sample2.new)
Proportion <- round((prop.table(Number)),3)
car.sample.biased.new <- addmargins(rbind(Number, Proportion))[1:2,]
car.sample.biased.new

# Expected value for NCAP rating from original population
pop <- car$ncap_rating
```

```r
p <- round(prop.table(table(ifelse(pop > 3, "more than 3", pop))),3)
expected <- rbind(p*100,p)
expected

df <- 4-1

# chi square test for random sampling
chi.unbiased <- sum(((car.sample.unbiased.new[1:4]-expected[1:4])^2)/(
   expected[1:4]))
chi.unbiased

pvalue.unbiased <- pchisq(chi.unbiased, df, lower.tail = FALSE)
pvalue.unbiased

# chi square test for biased sampling
chi.biased <- sum(((car.sample.biased.new[1:4]-expected[1:4])^2)/(
   expected[1:4]))
chi.biased

pvalue.biased <- pchisq(chi.biased, df, lower.tail = FALSE)
pvalue.biased

# chi test with R functions
chisq.test(car.sample.unbiased.new[1,1:4], p = expected[2,1:4])
chisq.test(car.sample.biased.new[1,1:4], p = expected[2,1:4])

#Part B ---------

ncap_rating <- car$ncap_rating
table(car$transmission_type,ncap_rating)

# observed
ncap_rating.new <- ifelse(ncap_rating > 2, "more than 2", ncap_rating)
contingency.table <- addmargins(table(car$transmission_type,ncap_
   rating.new))
contingency.table

# expected
expected<- contingency.table

numberOfcol <- ncol(expected)
numberOfrow <- nrow(expected)
table.total <- expected[numberOfrow,numberOfcol]

R <- 1:nrow(expected)
C <- 1:ncol(expected)

for (r in R) {
  for (c in C) {
    row.total   <- expected[r,numberOfcol]
    col.total   <- expected[numberOfrow,c]
    expected[r,c] <- round(row.total * col.total/ table.total)
  }
}

expected

```

```r
# chi square test
chi <- sum(((contingency.table[1:2,1:3] - expected[1:2,1:3])^2)/(
    expected[1:2,1:3 ]))
chi

df <- (length(R)-1) * (length(C)-1)

pvalue <- pchisq(chi, 2, lower.tail = FALSE)
pvalue

# chi test with R functions
chisq.test(contingency.table,  rescale.p = T)


#-----> Question 4 <-----
# Part A ---------

# response
policy <- car$policy_tenure

# explanatory
age <- car$age_of_car
displacement <- car$displacement

# Part B ---------
model.with.age <- lm(car$policy_tenure ~ car$age_of_car)
model.with.dis <- lm(car$policy_tenure ~ car$displacement)

# age of the car

# Sub Part a -----
# method 1
library(ggplot2)
library(ggfortify)
autoplot(model.with.age)
#--------

# method 2
#Linearity
plot(model.with.age,1)
#2.Nearly normal residuals
plot(model.with.age,2)
# 3.Constant variability
plot(model.with.age,3)

# Sub Part b -----
summary(model.with.age)

# Sub Part c -----


# Sub Part d -----

ggplot(car, aes(y=policy_tenure, x=age_of_car)) +
  geom_point( alpha = .2) +
  geom_smooth(method=lm, col = 'blue',se = FALSE,formula = 'y ~ x',
   linetype="dashed") +
```

```
  labs(title=paste("Scatter Plot of Age of car and policy tenure and
   fitted linear regression curve"))


# displacement

# Sub Part a -----
# method 1
autoplot(model.with.dis)
#--------

# method 2
#Linearity
plot(model.with.dis,1)
#2.Nearly normal residuals
plot(model.with.dis,2)
# 3.Constant variability
plot(model.with.dis,3)

# Sub Part b -----
summary(model.with.dis)

# Sub Part c -----


# Sub Part d -----

ggplot(car, aes(y=policy_tenure, x=displacement)) +
  geom_point( alpha = .2) +
  geom_smooth(method=lm, col = 'blue',se = FALSE,formula = 'y ~ x',
   linetype="dashed") +
  labs(title=paste("Scatter Plot of Age of car and policy tenure and
   fitted linear regression curve"))


# Part C ---------
# Part D ---------
anova(model.with.age)
anova(model.with.dis)

# Part E ---------
# Part F ---------
set.seed(1)
car_sample <- sample_n(car, 100)

train.car <- car_sample[1:90, ]
test.car <- car_sample[91:100, ]

# Sub Part a -----
model2.with.age<- lm(policy_tenure ~ age_of_car, train.car)
summary(model2.with.age)

model2.with.dis<- lm(policy_tenure ~ displacement, train.car)
summary(model2.with.dis)


```

```r
# Sub Part b -----
b1 <- model2.with.age$coefficients[2]
se1 <- summary(model2.with.age)$coefficient[4]
tstar <- abs(qt(0.025, df = 90-1-1))
me1 <- se1 * tstar

lowerinterval1 <- b1 - me1
lowerinterval1

upperinterval1 <- b1 + me1
upperinterval1


b2 <- model2.with.dis$coefficients[2]
se2 <- summary(model2.with.dis)$coefficient[4]
tstar <- abs(qt(0.025, df = 90-1-1))
me2 <- se2 * tstar

lowerinterval2 <- b2 - me2
lowerinterval2

upperinterval2 <- b2 + me2
upperinterval2

# Sub Part c -----
predict1 <- predict(model2.with.age,select(test.car,age_of_car))
predict2 <- predict(model2.with.dis,select(test.car,displacement))

# Sub Part d -----
actual1  <- select(test.car,policy_tenure)
diff1    <- round(abs(predict1-actual1),1)

actual2  <- select(test.car,policy_tenure)
diff2    <- round(abs(predict2-actual2),1)

success.rate1 <- length(diff1[diff1== 0])/nrow(diff1)
success.rate1

success.rate2 <- length(diff2[diff2== 0])/nrow(diff2)
success.rate2

#-----> Question 5 <-----
# Part A ---------
# choose all numerical var

data <- select_if(car, is.numeric)          # Identify numeric columns
data <- data[,!names(data) %in% c("X")]

#create pairs plot
ggpairs(data)


car.sample <- data %>% select(policy_tenure,width, displacement,ncap_
    rating,age_of_car,height)
ggpairs(car.sample)


```

```r
# Part B ---------
model_partB <- lm(policy_tenure ~ . , data = car.sample)
summary(model_partB)

# Part C ---------
summary(model)$r.squared

# Part D ---------

build_model = function(res, exp) {
  as.formula(paste(res, paste(exp, collapse=" + "), sep=" ~ "))
}

full.variables <- c("age_of_car","age_of_policyholder","population_density","airbags","displacement","cylinder","gear_box","turning_radius","length","width","height","gross_weight","ncap_rating","is_claim")

#-----> method 1 <-----
#backward elimination - adjusted r square
full.model <- lm(build_model('policy_tenure',full.variables)  , data = data)
summary(full.model) #0.0947

#step 1
#delete age_of_car
variables <- c("age_of_policyholder","population_density","airbags","displacement","cylinder","gear_box","turning_radius","length","width","height","gross_weight","ncap_rating","is_claim")
model.s11 <- lm(build_model('policy_tenure', variables)  , data = data)
summary(model.s11) # 0.09012

#delete age_of_policyholder
variables <- c("age_of_car","population_density","airbags","displacement","cylinder","gear_box","turning_radius","length","width","height","gross_weight","ncap_rating","is_claim")
model.s12 <- lm(build_model('policy_tenure', variables)  , data = data)
summary(model.s12) # 0.07158

#delete population_density
variables <- c("age_of_car","age_of_policyholder","airbags","displacement","cylinder","gear_box","turning_radius","length","width","height","gross_weight","ncap_rating","is_claim")
model.s13 <- lm(build_model('policy_tenure', variables)  , data = data)
summary(model.s13) # 0.08951

#delete airbags
variables <- c("age_of_car","age_of_policyholder","population_density","displacement","cylinder","gear_box","turning_radius","length","width","height","gross_weight","ncap_rating","is_claim")
model.s14 <- lm(build_model('policy_tenure', variables)  , data = data)
summary(model.s14) # 0.09467

```

```r
#delete displacement
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","cylinder","gear_box","turning_radius","length","width"
    ,"height","gross_weight","ncap_rating","is_claim")
model.s15 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s15) # 0.0971

#delete cylinder
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    width","height","gross_weight","ncap_rating","is_claim")
model.s16 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s16) # 0.09472

#delete gear_box
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","turning_radius","length","
    width","height","gross_weight","ncap_rating","is_claim")
model.s17 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s17) # 0.09436

#delete turning_radius
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","length","width","
    height","gross_weight","ncap_rating","is_claim")
model.s18 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s18) # 0.09326

#delete length
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","turning_radius","
    width","height","gross_weight","ncap_rating","is_claim")
model.s19 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s19) # 0.09465

#delete width
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","turning_radius","
    length","height","gross_weight","ncap_rating","is_claim")
model.s110 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s110) # 0.09418

#delete height
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","turning_radius","
    length","width","gross_weight","ncap_rating","is_claim")
model.s111 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s111) # 0.09374

#delete gross_weight
```

```
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","turning_radius","
    length","width","height","ncap_rating","is_claim")
model.s112 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s112) # 0.09454

#delete ncap_rating
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","turning_radius","
    length","width","height","gross_weight","is_claim")
model.s113 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s113) # 0.09464

#delete is_claim
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","cylinder","gear_box","turning_radius","
    length","width","height","gross_weight","ncap_rating")
model.s114 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s114) # 0.08925

### delete cylinder

#step 2
#delete age_of_car
variables <- c("age_of_policyholder","population_density","airbags","
    displacement","gear_box","turning_radius","length","width","height
    ","gross_weight","ncap_rating","is_claim")
model.s21 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s21) # 0.08971

#delete age_of_policyholder
variables <- c("age_of_car","population_density","airbags","
    displacement","gear_box","turning_radius","length","width","height
    ","gross_weight","ncap_rating","is_claim")
model.s22 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s22) # 0.07161

#delete population_density
variables <- c("age_of_car","age_of_policyholder","airbags","
    displacement","gear_box","turning_radius","length","width","height
    ","gross_weight","ncap_rating","is_claim")
model.s23 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s23) # 0.08953

#delete airbags
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"displacement","gear_box","turning_radius","length","width","
    height","gross_weight","ncap_rating","is_claim")
model.s24 <- lm(build_model('policy_tenure', variables)  , data = data
    )
summary(model.s24) # 0.09443
```

```
542
543 #delete displacement
544 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","gear_box","turning_radius","length","width","height","
    gross_weight","ncap_rating","is_claim")
545 model.s25 <- lm(build_model('policy_tenure', variables)  , data = data
    )
546 summary(model.s25) # 0.0973
547
548 #delete gear_box
549 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","turning_radius","length","width","
    height","gross_weight","ncap_rating","is_claim")
550 model.s27 <- lm(build_model('policy_tenure', variables)  , data = data
    )
551 summary(model.s27) # 0.09434
552
553 #delete turning_radius
554 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","length","width","height","
    gross_weight","ncap_rating","is_claim")
555 model.s28 <- lm(build_model('policy_tenure', variables)  , data = data
    )
556 summary(model.s28) # 0.09289
557
558 #delete length
559 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","width","
    height","gross_weight","ncap_rating","is_claim")
560 model.s29 <- lm(build_model('policy_tenure', variables)  , data = data
    )
561 summary(model.s29) # 0.09467
562
563 #delete width
564 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    height","gross_weight","ncap_rating","is_claim")
565 model.s210 <- lm(build_model('policy_tenure', variables)  , data =
    data)
566 summary(model.s210) # 0.09201
567
568 #delete height
569 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    width","gross_weight","ncap_rating","is_claim")
570 model.s211 <- lm(build_model('policy_tenure', variables)  , data =
    data)
571 summary(model.s211) # 0.08958
572
573 #delete gross_weight
574 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    width","height","ncap_rating","is_claim")
575 model.s212 <- lm(build_model('policy_tenure', variables)  , data =
    data)
576 summary(model.s212) # 0.09302
577
```

```r
#delete ncap_rating
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    width","height","gross_weight","is_claim")
model.s213 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s213) # 0.09384

#delete is_claim
variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    width","height","gross_weight","ncap_rating")
model.s214 <- lm(build_model('policy_tenure', variables)  , data =
    data)
summary(model.s214) # 0.08927

# backward Elimination choose model.s16
best.model.back <- model.s16

#-----> method 2 <-----
#forward selection -  adjusted r square

forward_select <- function(data, response) {
  predictors <- names(data)
  predictors <- predictors[!predictors %in% response]
  selected <- c()
  current_score <- -100

  for (i in 1:length(predictors)) {
    best_predictor <- NULL
    best_new_score <- -100

    for (predictor in setdiff(predictors, selected)) {
      model <- lm(as.formula(paste(response, paste(c(selected,
   predictor), collapse = " + "), sep = " ~ ")), data = data)
      new_score <- summary(model)$r.squared

      if (new_score > best_new_score) {
        best_predictor <- predictor
        best_new_score <- new_score
      }
    }

    if (best_new_score > current_score) {
      selected <- c(selected, best_predictor)
      current_score <- best_new_score
    } else {
      break
    }
  }

  return(selected)
}

best.variable <- forward_select(data, "policy_tenure")
best.model.for <- lm(build_model('policy_tenure', best.variable)  ,
    data = data)
```

```
627
628 #-----> method 3 <-----
629 #backward elimination - p value
630 full.model2 <- lm(build_model('policy_tenure',full.variables)  , data
    = data)
631 summary(full.model2) #0.0947
632
633 #step 1
634 #delete cylinder
635 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","displacement","gear_box","turning_radius","length","
    width","height","gross_weight","ncap_rating","is_claim")
636 model2.s11 <- lm(build_model('policy_tenure', variables)  , data =
    data)
637 summary(model2.s11) # 0.09472
638
639 #step 2
640 #delete displacement
641 variables <- c("age_of_car","age_of_policyholder","population_density"
    ,"airbags","gear_box","turning_radius","length","width","height","
    gross_weight","ncap_rating","is_claim")
642 model2.s12 <- lm(build_model('policy_tenure', variables)  , data =
    data)
643 summary(model2.s12) # 0.09473
644
645 best.model.back.pvalue <- model2.s12
646
647
648 #-----> method 4 <-----
649 modell <- lm(policy_tenure ~ ., data = data)
650 best.model.for.pvalue <- step(modell, direction = "forward", scope =
    formula(modell), trace = 0, k = log(nrow(data)))
651
652
653 #----->  Results <-----
654 summary(best.model.back) #method 1
655 summary(best.model.for) #method 2
656 summary(best.model.back.pvalue) #method 3
657 summary(best.model.for.pvalue) #method 4
658
659 # Part E ---------
660 library(caret)
661 train_control <- trainControl(method = "cv", number = 5)
662
663 # part B
664 predictors <- names(model_partB$coefficients)
665 predictors <- predictors[!predictors %in% '(Intercept)']
666
667 model.last <- train(build_model('policy_tenure', predictors), data =
    data , method = "lm",trControl = train_control)
668 model.last
669
670
671 # part D
672 predictors <- names(best.model.back.pvalue$coefficients)
673 predictors <- predictors[!predictors %in% '(Intercept)']
674
```

```
675 model.last2 <- train(build_model('policy_tenure', predictors), data =
       data , method = "lm",trControl = train_control)
676 model.last2
677
678 # Part F ---------
679 autoplot(best.model.back.pvalue)
680 plot(best.model.back.pvalue,1)
681 plot(best.model.back.pvalue,2)
682 plot(best.model.back.pvalue,3)
683 # Part G ---------
```