343_Data_Analysis_Project

Jiaming Hu Anqu Yu

12/2/2021

1. Data Preparation

Let's first read the data.

```
require(MASS)
```

```
## Loading required package: MASS
```

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

```
auto <- read.table("auto.txt", header = TRUE)
dim(auto)
```

```
## [1] 205  22
```

```
summary(auto)
```

```
##      make             fuel_type           wheel_base         length
##  Length:205         Length:205         Min.   : 86.60   Min.   :141.1
##  Class :character   Class :character   1st Qu.: 94.50   1st Qu.:166.3
##  Mode  :character   Mode  :character   Median : 97.00   Median :173.2
##                                        Mean   : 98.76   Mean   :174.0
##                                        3rd Qu.:102.40   3rd Qu.:183.1
##                                        Max.   :120.90   Max.   :208.1
##      width           height       curb_weight   num_of_doors
##  Min.   :60.30   Min.   :47.80   Min.   :1488   Length:205
##  1st Qu.:64.10   1st Qu.:52.00   1st Qu.:2145   Class :character
##  Median :65.50   Median :54.10   Median :2414   Mode  :character
##  Mean   :65.91   Mean   :53.72   Mean   :2556
##  3rd Qu.:66.90   3rd Qu.:55.50   3rd Qu.:2935
##  Max.   :72.30   Max.   :59.80   Max.   :4066
##   body_style        drive_wheels       num_of_cylinders   engine_type
##  Length:205         Length:205         Length:205         Length:205
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  fuel_system         aspiration         engine_size       bore
##  Length:205         Length:205         Min.   : 61.0   Length:205
##  Class :character   Class :character   1st Qu.: 97.0   Class :character
##  Mode  :character   Mode  :character   Median :120.0   Mode  :character
##                                        Mean   :126.9
##                                        3rd Qu.:141.0
##                                        Max.   :326.0
```

```
##     stroke           compression_rate   peak_rpm          horsepower
##  Length:205         Min.   : 7.00    Length:205         Length:205
##  Class :character   1st Qu.: 8.60    Class :character   Class :character
##  Mode  :character   Median : 9.00    Mode  :character   Mode  :character
##                     Mean   :10.14
##                     3rd Qu.: 9.40
##                     Max.   :23.00
##  normalized_losses   highway_mpg
##  Length:205         Min.   :16.00
##  Class :character   1st Qu.:25.00
##  Mode  :character   Median :30.00
##                     Mean   :30.75
##                     3rd Qu.:34.00
##                     Max.   :54.00
```

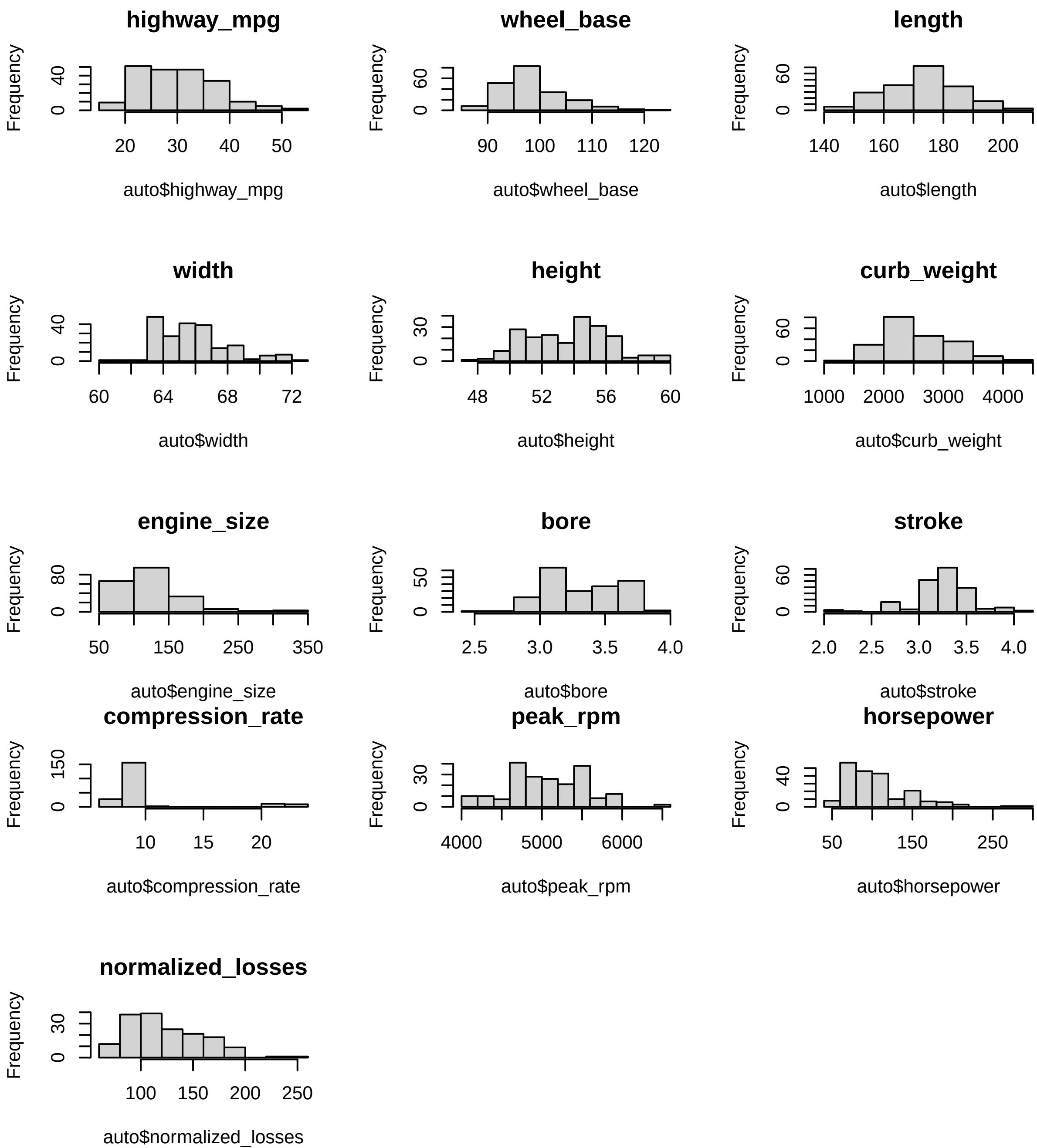
We need to tell `R` which groups of covariates are numerical and which are categorical, as most of the entries are characters in the summary shown above. According to the information provided, we know the properties of these covariates. we should change them into numbers or factors. Here is some steps to make clear whether every covariate is a categorical or numerical, according to the data description provided. For better dealing with the data, we will replace all the "?" missing data with '`NA` so that `R` can know they are missing.

```r
# put indecies of numeric or categorical covariates together for convenience
categorical <- c(1,2,8,9,10,11,12,13,14)
cate_var <- colnames(auto)[categorical]
numerical <- c(3,4,5,6,7,15,16,17,18,19,20,21)
numerical_var <- colnames(auto)[numerical]

# convert "?" into NA as it is easier to deal with in R
for(i in 1:dim(auto)[1]){
  for(j in 1:dim(auto)[2]){
    if(auto[i,j]=="?"){
      auto[i,j] <- NA
    }
  }
}

# change classification of entries
for(i in 1:length(categorical)){
  auto[,categorical[i]] <- as.factor(auto[,categorical[i]])
}
for(i in 1:(length(numerical))){
  auto[,numerical[i]] <- as.numeric(auto[,numerical[i]])
}
```

Let's check the data summary again and look at distribution.

```
##          make       fuel_type     wheel_base         length          width
##  toyota    : 32   diesel: 20   Min.   : 86.60   Min.   :141.1   Min.   :60.30
##  nissan    : 18   gas   :185   1st Qu.: 94.50   1st Qu.:166.3   1st Qu.:64.10
##  mazda     : 17                Median : 97.00   Median :173.2   Median :65.50
##  honda     : 13                Mean   : 98.76   Mean   :174.0   Mean   :65.91
##  mitsubishi: 13                3rd Qu.:102.40   3rd Qu.:183.1   3rd Qu.:66.90
##  subaru    : 12                Max.   :120.90   Max.   :208.1   Max.   :72.30
##  (Other)   :100
##      height       curb_weight   num_of_doors       body_style drive_wheels
##  Min.   :47.80   Min.   :1488   four:114     convertible: 6   4wd:  9
```

```
##  1st Qu.:52.00   1st Qu.:2145   two : 89     hardtop    : 8   fwd:120
##  Median :54.10   Median :2414   NA's:  2     hatchback  :70   rwd: 76
##  Mean   :53.72   Mean   :2556                sedan      :96
##  3rd Qu.:55.50   3rd Qu.:2935                wagon      :25
##  Max.   :59.80   Max.   :4066
##
##  num_of_cylinders engine_type  fuel_system aspiration   engine_size
##  eight :  5       dohc : 12   mpfi   :94   std  :168   Min.   : 61.0
##  five  : 11       dohcv:  1   2bbl   :66   turbo: 37   1st Qu.: 97.0
##  four  :159       l    : 12   idi    :20               Median :120.0
##  six   : 24       ohc  :148   1bbl   :11               Mean   :126.9
##  three :  1       ohcf : 15   spdi   : 9               3rd Qu.:141.0
##  twelve:  1       ohcv : 13   4bbl   : 3               Max.   :326.0
##  two   :  4       rotor:  4   (Other): 2
##       bore          stroke      compression_rate    peak_rpm      horsepower
##  Min.   :2.54   Min.   :2.070   Min.   : 7.00    Min.   :4150   Min.   : 48.0
##  1st Qu.:3.15   1st Qu.:3.110   1st Qu.: 8.60    1st Qu.:4800   1st Qu.: 70.0
##  Median :3.31   Median :3.290   Median : 9.00    Median :5200   Median : 95.0
##  Mean   :3.33   Mean   :3.255   Mean   :10.14    Mean   :5125   Mean   :104.3
##  3rd Qu.:3.59   3rd Qu.:3.410   3rd Qu.: 9.40    3rd Qu.:5500   3rd Qu.:116.0
##  Max.   :3.94   Max.   :4.170   Max.   :23.00    Max.   :6600   Max.   :288.0
##  NA's   :4      NA's   :4                        NA's   :2      NA's   :2
##  normalized_losses  highway_mpg
##  Min.   : 65       Min.   :16.00
##  1st Qu.: 94       1st Qu.:25.00
##  Median :115       Median :30.00
##  Mean   :122       Mean   :30.75
##  3rd Qu.:150       3rd Qu.:34.00
##  Max.   :256       Max.   :54.00
##  NA's   :41
```



The continuous data distributions look fine. The `wheel_base`, `horsepower`, `ompassion_rate` and `normalized_losses` are a little skewed, we should keep this in mind in later steps, and consider transformation if necessary. Also notice that there are small numbers of missing data in several covariates, and a lot of missing data in `normalized_losses`. We will deal with this in next section.

2. Dealing with Collinearity and Missing Data

Since we have missing values, we need to deal with them before we do any analysis. Let's first see the number of samples that have missing data, with respect to each covariates.

```
colSums(is.na(auto))
```

```
##              make          fuel_type         wheel_base             length
##                 0                  0                  0                  0
##             width             height        curb_weight       num_of_doors
##                 0                  0                  0                  2
##        body_style       drive_wheels   num_of_cylinders        engine_type
##                 0                  0                  0                  0
##       fuel_system         aspiration        engine_size               bore
##                 0                  0                  0                  4
##            stroke   compression_rate           peak_rpm         horsepower
##                 4                  0                  2                  2
## normalized_losses        highway_mpg
##                41                  0
```

We can see that `num_of_doors`, `bore`, `stroke`, `peak_rpm`, `horsepower` and `normalized_losses` have missing data. Observe that besides `normalized_losses`, other covariates that have missing values are only missing in a relatively small number of entries, relative to sample size, since we have 201 sample, so we can simply delete those samples. As for `normalized_losses`, we first try imputation by mean because deleting ~20% (41) of the sample is not ideal.

```
index <- NULL
for(i in 1:dim(auto)[1]){
  for(j in 1:20){
    if(is.na(auto[i,j])){
      index <- c(index,i)
    }
  }
}
index <- unique(index)
length(index)
```

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

So we only have 8 samples to delete, this is acceptable. Now look at `normalized_losses`, which has 41 missing entries.

```
auto <- auto[-index,]
colSums(is.na(auto))
```

```
##              make          fuel_type         wheel_base             length
##                 0                  0                  0                  0
##             width             height        curb_weight       num_of_doors
##                 0                  0                  0                  0
##        body_style       drive_wheels   num_of_cylinders        engine_type
##                 0                  0                  0                  0
##       fuel_system         aspiration        engine_size               bore
##                 0                  0                  0                  0
##            stroke   compression_rate           peak_rpm         horsepower
##                 0                  0                  0                  0
## normalized_losses        highway_mpg
##                38                  0
```

```r
# find the indeces of entries with missing normalized_losses
index_imp <- NULL
for(i in 1:dim(auto)[1]){
  for(j in 1:21){
    if(is.na(auto[i,j])){
      index_imp <- c(index_imp,i)
    }
  }
}
index_imp <- unique(index_imp)
# make a subset data without any missing data or imputed data
subset_auto <- auto[-index_imp, ]
```

We have 38 samples missing the `normalized_losses` entries. Either imputing the mean or regression should be applied in this case, but since we have 21 covariates, imputing with regression may be time-consuming and not feasible. After consideration, we decide to check the collinearity of the numerical variables before doing any imputation for `normalized_losses`. Since the variables are all related to cars, many of them might have correlations.

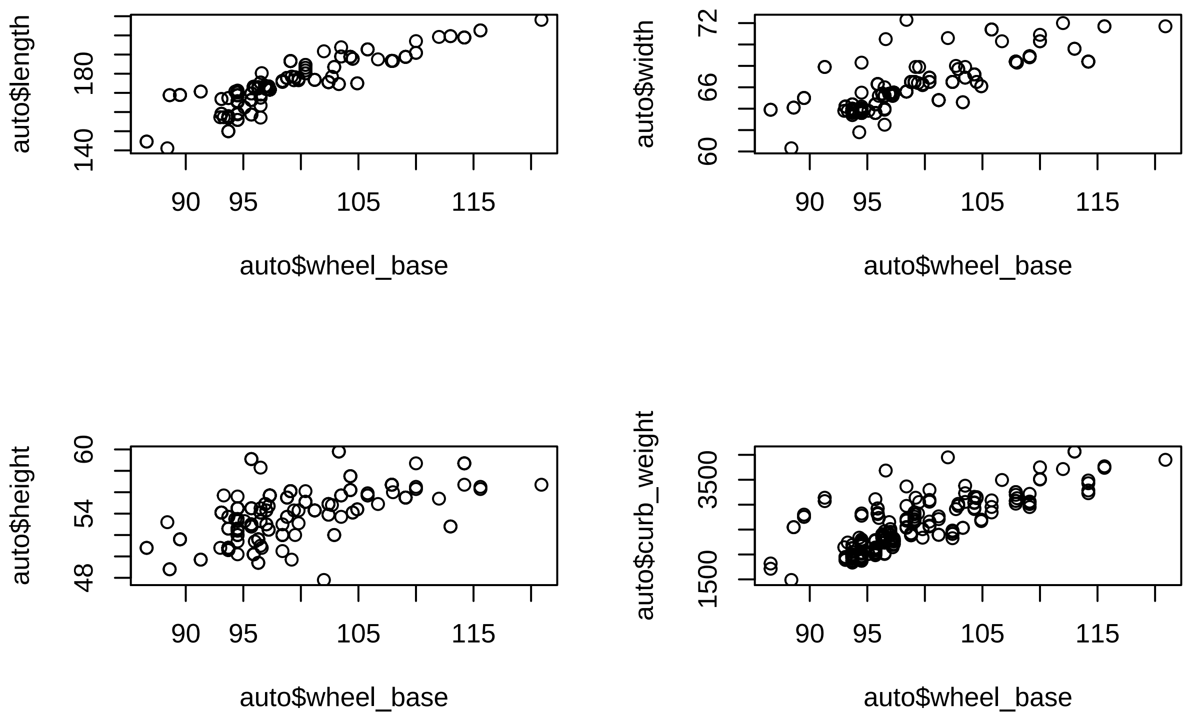
```r
round(cor(auto[numerical],use="complete.obs"),3)
```

```
##                   wheel_base length  width height curb_weight engine_size
## wheel_base             1.000  0.872  0.815  0.556       0.810       0.649
## length                 0.872  1.000  0.838  0.499       0.871       0.726
## width                  0.815  0.838  1.000  0.293       0.871       0.779
## height                 0.556  0.499  0.293  1.000       0.367       0.111
## curb_weight            0.810  0.871  0.871  0.367       1.000       0.889
## engine_size            0.649  0.726  0.779  0.111       0.889       1.000
## bore                   0.578  0.646  0.573  0.255       0.646       0.596
## stroke                 0.167  0.121  0.197 -0.091       0.174       0.300
## compression_rate       0.291  0.185  0.259  0.233       0.225       0.141
## peak_rpm              -0.289 -0.234 -0.232 -0.246      -0.260      -0.285
## horsepower             0.517  0.672  0.682  0.034       0.790       0.812
## normalized_losses     -0.060  0.036  0.110 -0.414       0.126       0.208
##                     bore stroke compression_rate peak_rpm horsepower
## wheel_base         0.578  0.167            0.291   -0.289      0.517
## length             0.646  0.121            0.185   -0.234      0.672
## width              0.573  0.197            0.259   -0.232      0.682
## height             0.255 -0.091            0.233   -0.246      0.034
## curb_weight        0.646  0.174            0.225   -0.260      0.790
## engine_size        0.596  0.300            0.141   -0.285      0.812
## bore               1.000 -0.103            0.015   -0.312      0.560
## stroke            -0.103  1.000            0.244   -0.011      0.149
## compression_rate   0.015  0.244            1.000   -0.417     -0.162
## peak_rpm          -0.312 -0.011           -0.417    1.000      0.074
## horsepower         0.560  0.149           -0.162    0.074      1.000
## normalized_losses -0.032  0.063           -0.127    0.238      0.291
##                   normalized_losses
## wheel_base                   -0.060
## length                        0.036
## width                         0.110
## height                       -0.414
## curb_weight                   0.126
## engine_size                   0.208
```

```
## bore                           -0.032
## stroke                          0.063
## compression_rate               -0.127
## peak_rpm                        0.238
## horsepower                      0.291
## normalized_losses               1.000
```

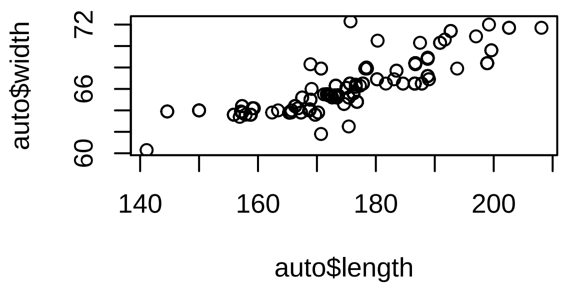
There are some interesting things to note: (1) the covariate needs imputing, `normalized_losses`, seems to have little correlation with other numerical variables, with only one correlation larger than 0.4. (2) there is obvious collinearity among more than one covariates, which we should check further.

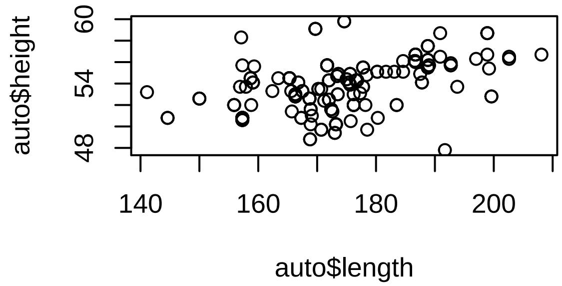
```
par(mfrow=c(2,2))
plot(auto$wheel_base, auto$length)
plot(auto$wheel_base, auto$width)
plot(auto$wheel_base, auto$height)
plot(auto$wheel_base, auto$curb_weight)
```

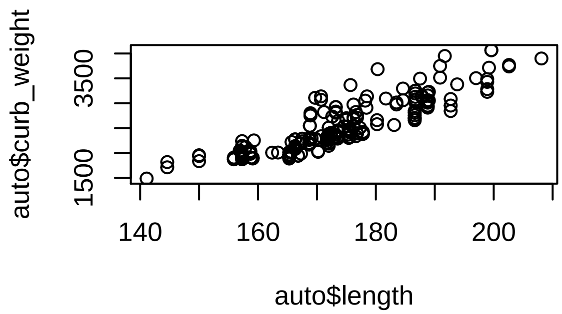


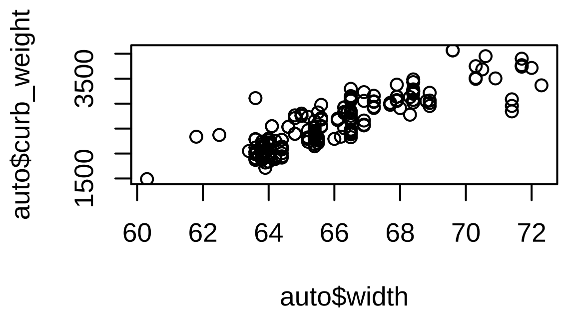
`wheel_base` has correlation with four other covariates, so we will remove it when modeling.

```
par(mfrow=c(2,2))
plot(auto$length, auto$width)
plot(auto$length, auto$height)
plot(auto$length, auto$curb_weight)


plot(auto$width, auto$curb_weight)
```

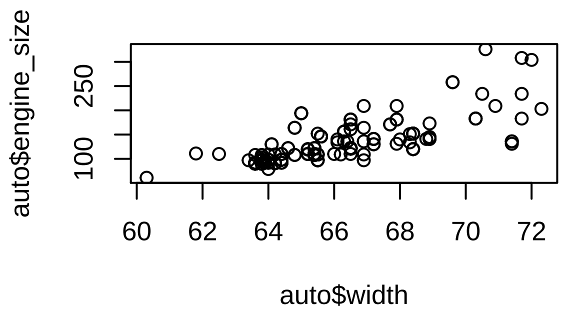


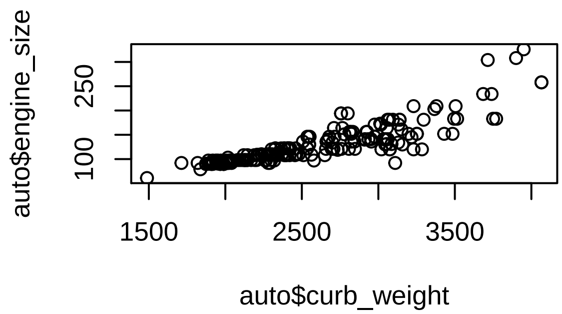






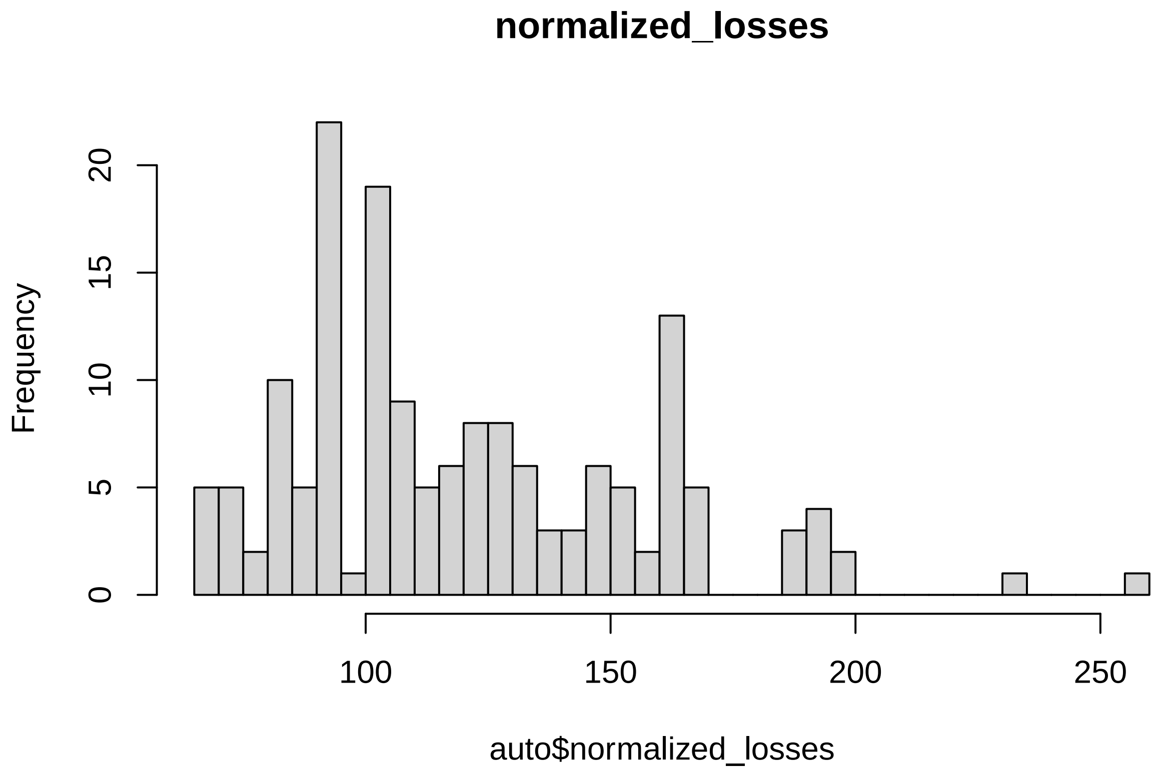

```
plot(auto$width, auto$engine_size)

plot(auto$curb_weight, auto$engine_size)
```





Similarly, some other covariates show positive correlation with each other. What we are showing here is a severe multi-collinearity issue among `wheel_base`, `length`, `width`, `curb_weight`,`height`. We will pay attention to this issue when doing regression.

```
hist(auto$normalized_losses, main="normalized_losses", breaks=40, xlim=c(65,256))
```



Another thing worth to note is that `normalized_losses` only has obvious correlationship with `height`. Also, the histogram of `normalized_losss` looks a little skewed, with a large range and some extreme values. Imputing with mean may not be ideal for this data, but we will first impute by mean and proceed to modeling, then go back and check if necessary.

```
nl_mean <- mean(as.numeric(na.omit(auto[,21])))
for(i in 1:dim(auto)[1]){
  if(is.na(auto[i,21])){
    auto[i,21] <- nl_mean
  }
}
```

3. Initial Regression

```
# an initial model with all covariates without interaction
auto$normalized_losses <- as.numeric(auto$normalized_losses)
lmod <- lm(highway_mpg~make+fuel_type+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses+height+wheel_base
           +length+width+height+curb_weight,auto)
round(summary(lmod)$coefficients[45:55,],3)
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## stroke              -0.205      1.387  -0.148    0.882
## compression_rate     1.590      0.664   2.395    0.018
## bore                -2.939      2.127  -1.382    0.169
## peak_rpm            -0.003      0.001  -3.211    0.002
## horsepower           0.025      0.030   0.842    0.401
## normalized_losses   -0.004      0.010  -0.379    0.706
## height              -0.219      0.202  -1.084    0.280
## wheel_base          -0.165      0.123  -1.344    0.181
## length              -0.123      0.068  -1.804    0.073
## width                0.609      0.305   1.999    0.047
## curb_weight         -0.006      0.002  -2.501    0.014
```

Missing data (again) in normalized_losses

As we can see in the summary, `the normalized_losses` is not a significant covariate. We should look more on this, because if it is truly not significantly affecting the response, we do not need to try imputing with regression.

```
lmod_less_data <- lm(highway_mpg~., data = subset_auto)
summary(lmod_less_data)$coefficient[42:49,]
```

```
##                        Estimate  Std. Error      t value     Pr(>|t|)
## aspirationturbo   -1.3475957222 1.219492322 -1.105046500 0.2715506328
## engine_size       -0.0002305926 0.049117879 -0.004694677 0.9962627068
## bore              -0.3295999239 3.000099389 -0.109863002 0.9127182073
## stroke            -1.6430092563 1.986009178 -0.827291875 0.4098627458
## compression_rate   1.0890087781 0.781892235  1.392786281 0.1664930961
## peak_rpm          -0.0041255445 0.001112044 -3.709875856 0.0003268982
## horsepower        -0.0371444797 0.040122327 -0.925780796 0.3565865507
## normalized_losses -0.0032035014 0.013127569 -0.244028529 0.8076632904
```

We can see that `normalized_losses` does not have a significant coefficient even without any imputed data and the estimate is very small. Although there is a small chance that all the missing entries can explain the effect of this term on the response, it is not practical for us to impute it and trust the data (if it changes with imputed regression entries). Also, `normalized_losses` is a quantitative measure of how much insurance companies pay for losses on this car. It is reasonable that it has almost nothing to do with miles-per-gallon performance. Therefore we will no longer discuss on the missing data issue, but keep the imputation with the mean and focus on optimizing the model.

Collinearity

Addressing the collinearity issue mentioned in the first part, we will only keep `curb_weight` out of the 5 correlated covariates (`wheel_base`, `length`, `width`, `curb_weight`, `height`) because it is the most significant in terms of p-values.

```r
lmod1 <- lm(highway_mpg~make+fuel_type+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses+curb_weight, auto)
round(summary(lmod1)$coefficients[45:51,],3)
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## stroke              -0.717      1.407  -0.510    0.611
## compression_rate     1.936      0.666   2.908    0.004
## bore                -2.786      2.153  -1.294    0.198
## peak_rpm            -0.003      0.001  -3.149    0.002
## horsepower           0.046      0.030   1.529    0.128
## normalized_losses    0.004      0.010   0.405    0.686
## curb_weight         -0.009      0.002  -5.005    0.000
```

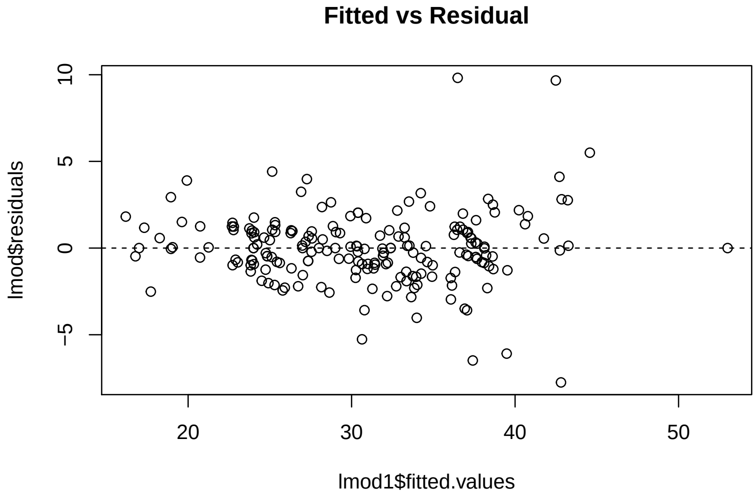
```r
anova(lmod1)
```

```
## Analysis of Variance Table
##
## Response: highway_mpg
##                    Df Sum Sq Mean Sq  F value    Pr(>F)
## make               20 4397.1  219.85  34.1631 < 2.2e-16 ***
## fuel_type           1  849.1  849.06 131.9361 < 2.2e-16 ***
## num_of_doors        1    0.9    0.88   0.1363 0.7125498
## body_style          4   69.1   17.29   2.6860 0.0336677 *
## drive_wheels        2 1009.1  504.57  78.4049 < 2.2e-16 ***
## num_of_cylinders    5  584.5  116.90  18.1644 5.458e-14 ***
## engine_type         4   76.3   19.09   2.9657 0.0216407 *
## fuel_system         5  727.6  145.52  22.6126 < 2.2e-16 ***
## aspiration          1  143.4  143.37  22.2784 5.474e-06 ***
## stroke              1   83.0   83.03  12.9020 0.0004478 ***
## compression_rate    1  115.7  115.70  17.9780 3.946e-05 ***
## bore                1   83.5   83.53  12.9804 0.0004309 ***
## peak_rpm            1   56.1   56.10   8.7178 0.0036734 **
## horsepower          1    7.5    7.54   1.1710 0.2809698
## normalized_losses   1    1.0    0.98   0.1518 0.6974290
## curb_weight         1  161.2  161.18  25.0459 1.589e-06 ***
## Residuals         146  939.6    6.44
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Residual plot

Lets see plots of fitted value vs residuals plot.

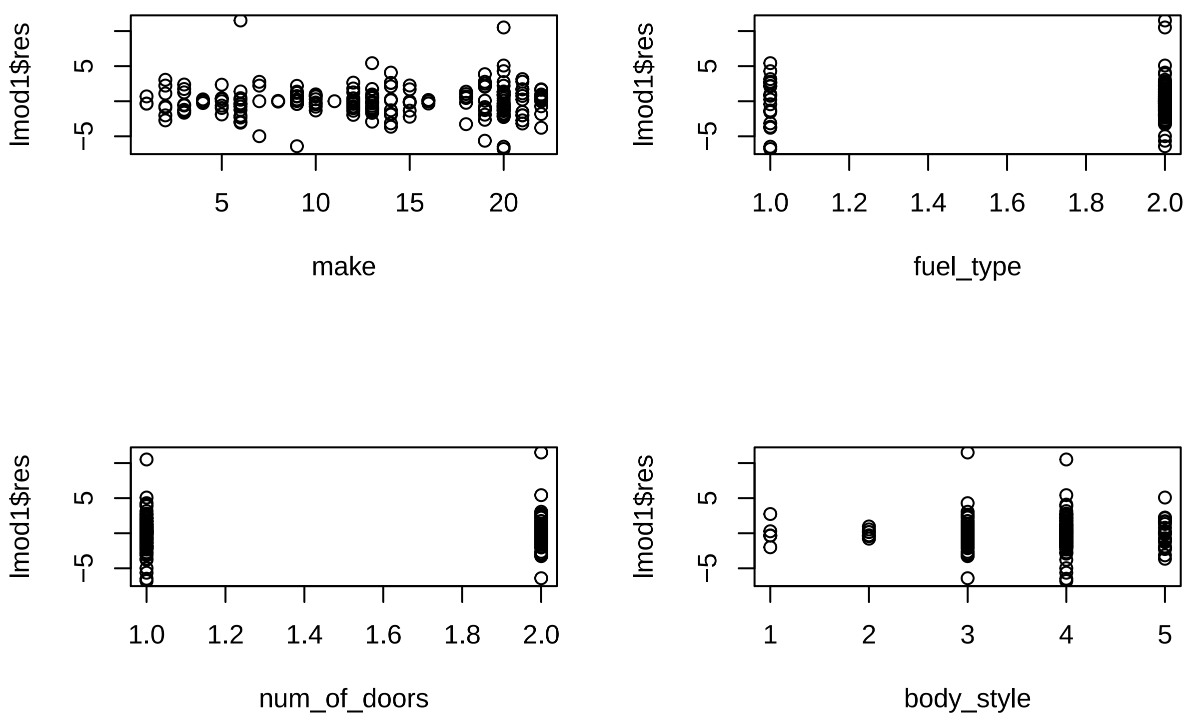
```r
plot(lmod1$fitted.values, lmod$residuals, main="Fitted vs Residual")
abline(h = 0, lty=2)
```



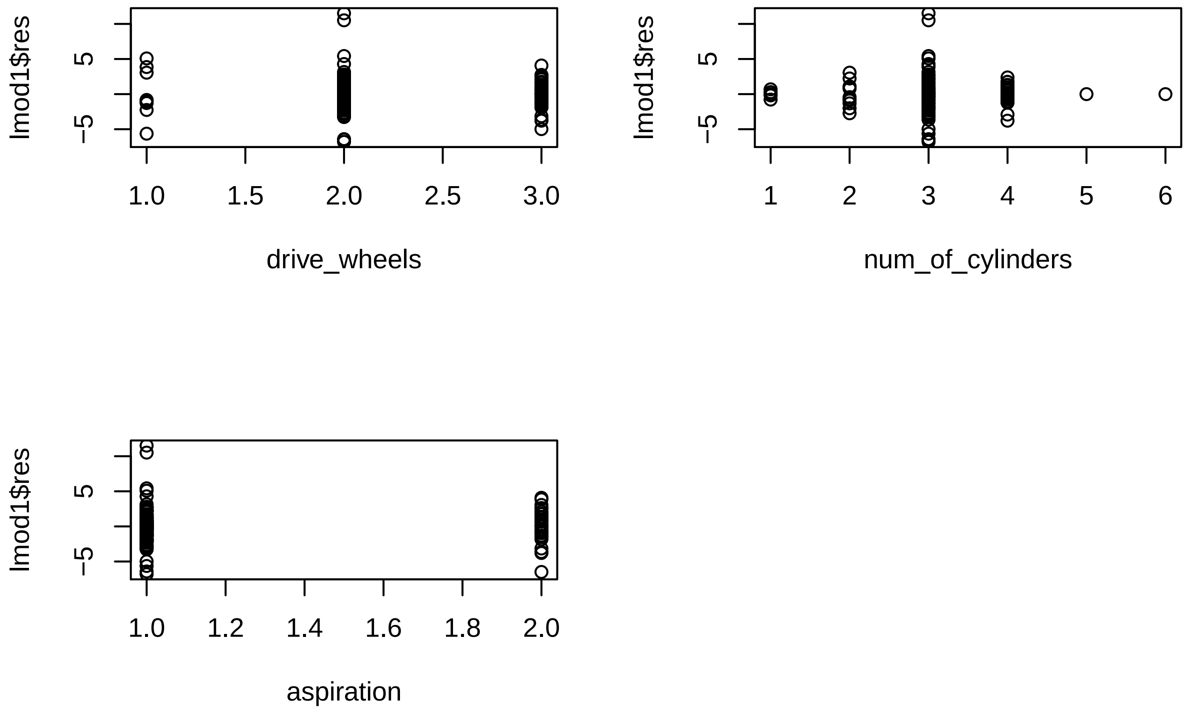
We perhaps can see a little bit of curvature and appearance of non-constant variance. There are two outliers, perhaps leverage points on top of the plot, and 3 on the bottom of the plot. We can use studentized residual and leverage score to see if they need to be deleted later. Also, it might be the case which we have more negative residuals than positive residuals. This mayy suggest we should do transformation as well.

Let's see categorical variable vs residual plots.

```
par(mfrow=c(2,2))
plot(as.numeric(auto$make),lmod1$res,xlab='make')
plot(as.numeric(auto$fuel_type),lmod1$res,xlab='fuel_type')
plot(as.numeric(auto$num_of_doors),lmod1$res,xlab='num_of_doors')
plot(as.numeric(auto$body_style),lmod1$res,xlab='body_style')
```

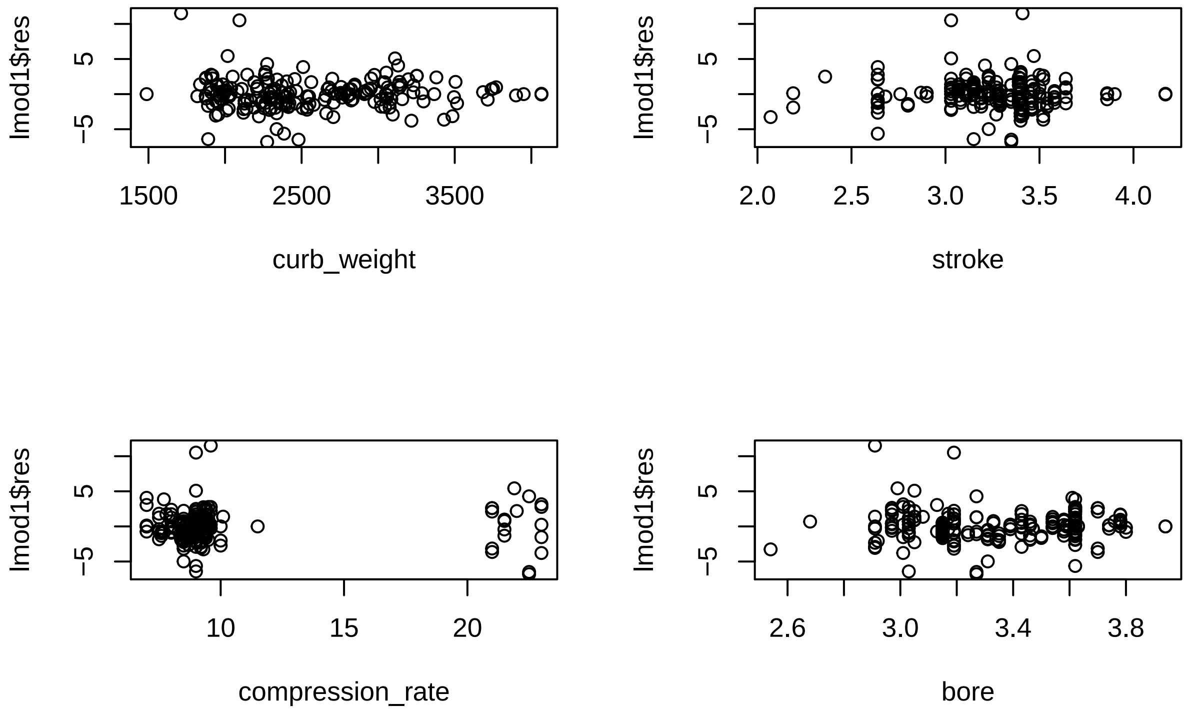



```
plot(as.numeric(auto$drive_wheels),lmod1$res,xlab='drive_wheels')
plot(as.numeric(auto$num_of_cylinders),lmod1$res,xlab='num_of_cylinders')
plot(as.numeric(auto$aspiration),lmod1$res,xlab='aspiration')
```

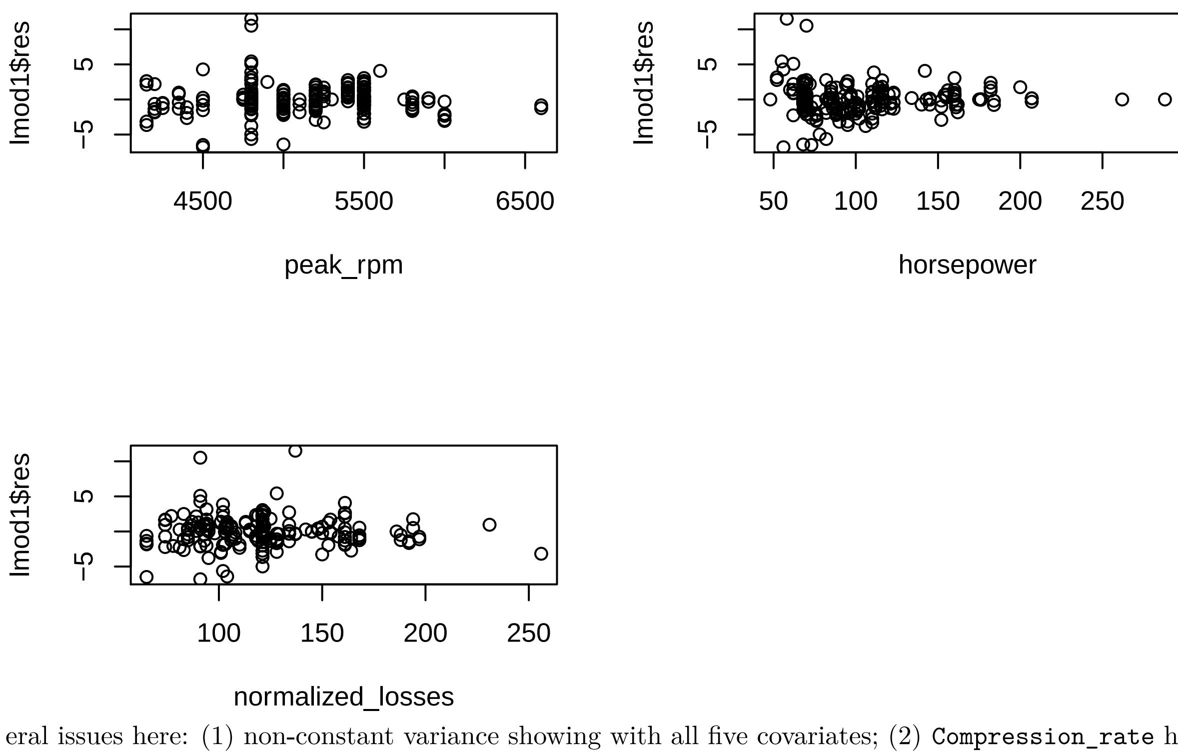


The residuals vs categorical covariate plots show some non-constant variant issue, but at this point we cannot tell whether the large variance in some covariates (e.g.`num_of_cylinders`) is because of large sample size at specific levels, or it is really a non-constant variance issue. Keeping this in mind, let's now see the residual plots against continuous covariates.

```
par(mfrow=c(2,2))
plot(as.numeric(auto$curb_weight),lmod1$res,xlab='curb_weight')
plot(as.numeric(auto$stroke),lmod1$res,xlab='stroke')
plot(as.numeric(auto$compression_rate),lmod1$res,xlab='compression_rate')
plot(as.numeric(auto$bore),lmod1$res,xlab='bore')
```




```r
plot(as.numeric(auto$peak_rpm),lmod1$res,xlab='peak_rpm')
plot(as.numeric(auto$horsepower),lmod1$res,xlab='horsepower')
plot(as.numeric(auto$normalized_losses),lmod1$res,xlab='normalized_losses')
```



Several issues here: (1) non-constant variance showing with all five covariates; (2) `Compression_rate` has two clusters. Since we have removed collinear covariates, we need to check for outliers and high-leverage points, then remove non-significant covariates, try seperate clusters of `compression_rate` if possible, and also try transforming data.

4. Diagnosis

Outliers

There are 2 values which contains NA in their studentized residuals, we can simply choose to omit them as the size of them is relatively small.

```
# any outliers?

max(abs(na.omit(studres(lmod1))))
```

```
## Warning in sqrt((n - p - sr^2)/(n - p - 1)): NaNs produced
```

```
## [1] 5.735299
```

```
threshold <- qt(1-0.05/2/(dim(auto)[1]-6),df=lmod1$df[1])
threshold
```

```
## [1] 3.742001
```

```
# how many NA's
sum(is.na(abs(studres(lmod1))))
```

```
## Warning in sqrt((n - p - sr^2)/(n - p - 1)): NaNs produced
```

```
## [1] 7
```

```
# There are 7 NA in studentized residuals, let's observe why it is the case or
# whether we need to worry about those two points

na.index <- as.numeric(which(is.na(studres(lmod1))))
```

```
## Warning in sqrt((n - p - sr^2)/(n - p - 1)): NaNs produced
```

```
lmod1$residuals[na.index]
```

```
##            19            30            47            50            76
##  8.461981e-15  3.105155e-15  9.225259e-15 -4.943962e-15  1.190367e-14
##           126           130
## -4.402728e-15 -2.404327e-15
```

```
# We can see it is possibly because the residuals are too small so when we use
# the formula in studentized residuals, they underflow. Hence we do not need to
# worry about those two points when detecting outliers.
outlier <- as.numeric(which(abs(studres(lmod1))>threshold))
```

```
## Warning in sqrt((n - p - sr^2)/(n - p - 1)): NaNs produced
```

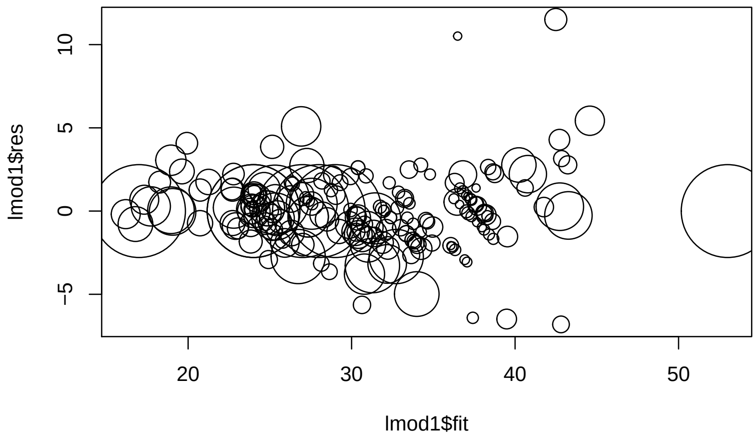
```
outlier
```

```
## [1]  30 153
```

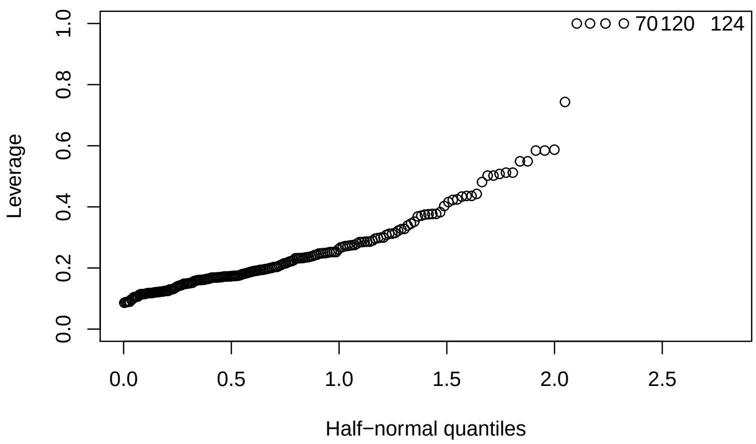
Leverage Point

Note here we use `hatvalues()` function to calculate the leverage score because our matrix model is singular. So we will use this instead of using the method that involves `t(x)%*%x`, but they serve the same purpose nontheless. We will investigate this problem in details when removing covariates (we put it in this order because it is more natural to solve it later instead of now, because it involves using F-test to compare models.)

```
plot(lmod1$fit,lmod1$res,cex=10*hatvalues(lmod1))
```




```
halfnorm(hatvalues(lmod1), ylab="Leverage", nlab=3)
```



We can see that in the first plot, the points at top and the points at bottom have low leverage score. This is good because we do not have to worry too much about them as high-leverage points. In the case that they are outliers, we had already calculated above, and found the outlier observations are #30 and #153.

On the half-norm plot of hatvalues, there are seven points with very high leverage. Look further:

```
hatv <- hatvalues(lmod1)
hatv[c(120,124,70)]
```

```
## 126 130  76
##   1   1   1
```

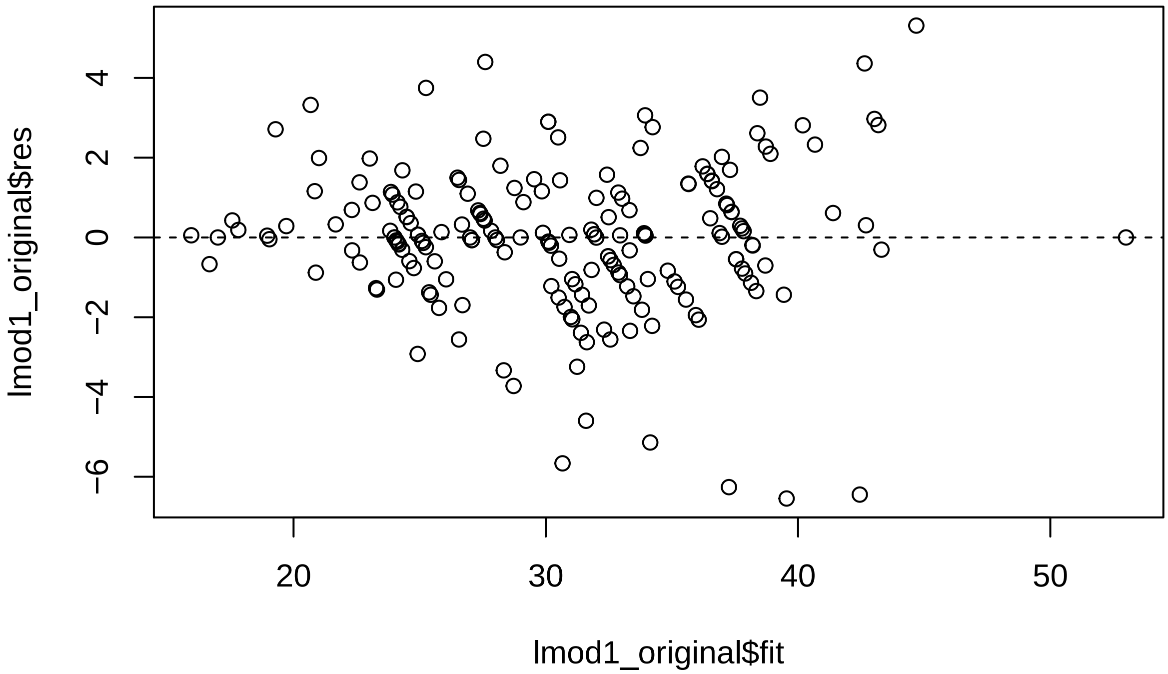
```
# check these observations and their entries for continuous covariates
auto[c(120,124,70), c(22,7,17,18,19,21)]
```

```
##     highway_mpg curb_weight stroke compression_rate peak_rpm normalized_losses
## 126          27        2778   3.11              9.5     5500          186.0000
## 130          28        3366   3.11             10.0     5750          121.1321
## 76           24        2910   3.12              8.0     5000          121.1321
```

Checking their means with the sample means, these entries look fine, without obvious extreme values. So we will keep them here for now. Before moving toward next steps, we will

```
# remove outliers
auto <- auto[-outlier,]

# fit the model again
```

```r
lmod1_original <- lm(highway_mpg~curb_weight+make+fuel_type+
               num_of_doors+body_style+drive_wheels+num_of_cylinders+
               engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
               peak_rpm+horsepower+normalized_losses, auto)
plot(lmod1_original$fit,lmod1_original$res)
abline(h = 0, lty=2)
```

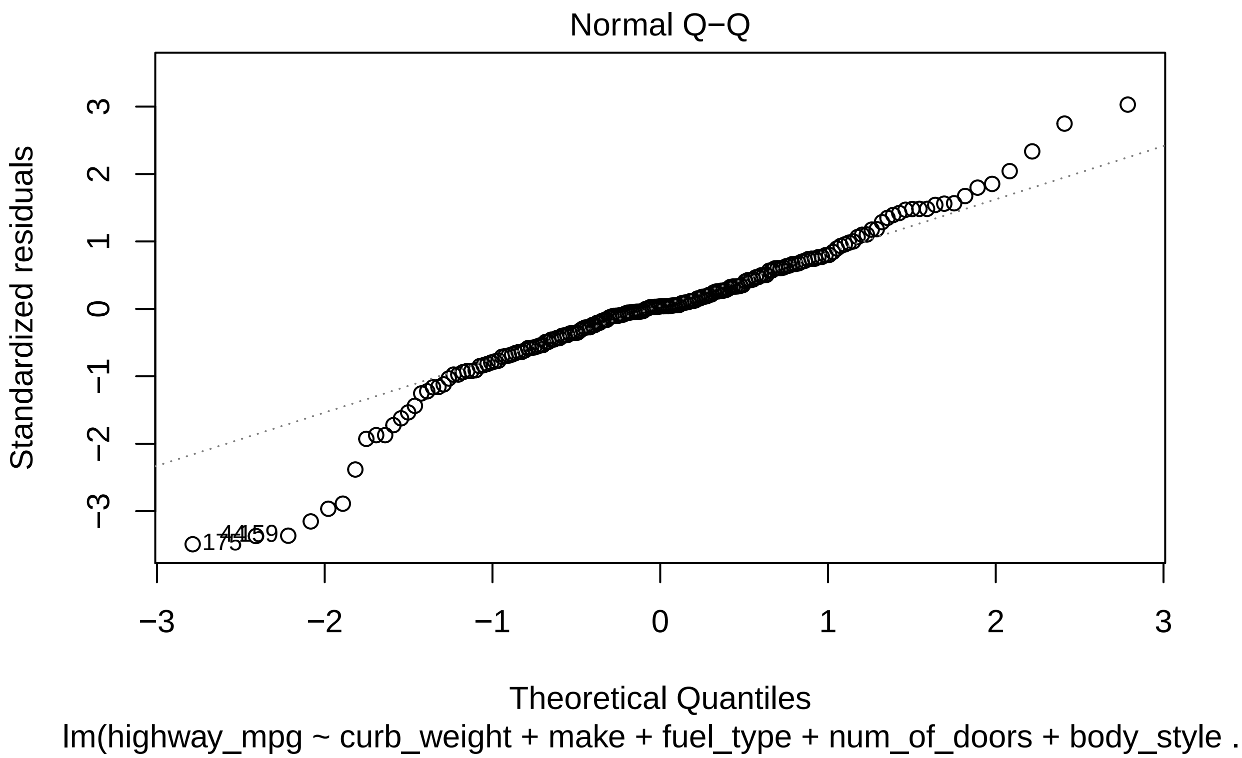


We can see the two points on the top has been removed. The residual plot looks better, although the residuals still show some non-constant variance issue.

Q-Q plot

```r
plot(lmod1_original, which=2); shapiro.test(lmod1_original$residuals)
```

```
## Warning: not plotting observations with leverage one:
##   19, 29, 45, 48, 69, 119, 123
```




```
##
##  Shapiro-Wilk normality test
##
## data:  lmod1_original$residuals
## W = 0.95271, p-value = 4.522e-06
```

The Q-Q plot here suggests that our model is not following normal distribution assumption well, and the Shapiro normality test suggests the violation of normal distribution. Since the normality assumption is not holding, we should transform covariates after removing non-significant covariates.

Conclusion after initial regression before transformation (1) 2 potential outliers at bottom of residual points (2) 1 potential leverage point to with large x value (3) There is appearance of non-constant variance (4) Violation in normality assumption (5) `num_of_cylinder` seems to be ordinal, maybe we could treat it as quantitative variable (6) Need to reduce model size and add interaction term if necessary

5. Remove non-significant Covariates (Backward Elimination)

```
round(summary(lmod1_original)$coef[22:51,4],3)
```

```
##             makevolvo          fuel_typegas        num_of_doorstwo
##                 0.327                 0.019                  0.671
##      body_stylehardtop   body_stylehatchback        body_stylesedan
##                 0.700                 0.683                  0.808
##        body_stylewagon       drive_wheelsfwd        drive_wheelsrwd
##                 0.969                 0.497                  0.794
##   num_of_cylindersfive  num_of_cylindersfour    num_of_cylinderssix
##                 0.877                 0.342                  0.673
##  num_of_cylindersthree num_of_cylinderstwelve      engine_typedohcv
##                 0.006                 0.022                  0.400
##         engine_typeohc         engine_typeohcf         engine_typeohcv
```

```
##                0.438               0.599               0.602
##       fuel_system2bbl        fuel_systemmfi       fuel_systemmpfi
##                0.121               0.101               0.041
##       fuel_systemspdi       fuel_systemspfi      aspirationturbo
##                0.037               0.040               0.018
##                stroke      compression_rate                 bore
##                0.918               0.002               0.086
##              peak_rpm            horsepower     normalized_losses
##                0.172               0.235               0.667
```

```r
# make all the models removing categorical covariates one by one
no.make <- lm(highway_mpg~curb_weight+fuel_type+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.fuel_type <- lm(highway_mpg~curb_weight+make+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.num_of_doors <- lm(highway_mpg~curb_weight+make+fuel_type+
              body_style+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.body_style <- lm(highway_mpg~curb_weight+make+fuel_type+
              num_of_doors+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.drive_wheels <- lm(highway_mpg~curb_weight+make+fuel_type+
              num_of_doors+body_style+num_of_cylinders+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.num_of_cylinders <- lm(highway_mpg~curb_weight+make+fuel_type+
              num_of_doors+body_style+drive_wheels+
              engine_type+fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.engine_type <- lm(highway_mpg~curb_weight+make+fuel_type+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              fuel_system+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.fuel_system <- lm(highway_mpg~curb_weight+make+fuel_type+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              engine_type+aspiration+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)
no.aspiration <- lm(highway_mpg~curb_weight+make+fuel_type+
              num_of_doors+body_style+drive_wheels+num_of_cylinders+
              engine_type+fuel_system+stroke+compression_rate+bore+
              peak_rpm+horsepower+normalized_losses, auto)

# check the performance of these models
round(anova(lmod1_original,no.make)$Pr[2],3)
```

```
## [1] 0.029
```

```r
round(anova(lmod1_original,no.fuel_type)$Pr[2],3)
```

```
## [1] NA
```

```r
round(anova(lmod1_original,no.num_of_doors)$Pr[2],3)
```

```
## [1] 0.671
```

```r
round(anova(lmod1_original,no.body_style)$Pr[2],3)
```

```
## [1] 0.864
```

```r
round(anova(lmod1_original,no.drive_wheels)$Pr[2],3)
```

```
## [1] 0.464
```

```r
round(anova(lmod1_original,no.num_of_cylinders)$Pr[2],3)
```

```
## [1] 0.007
```

```r
round(anova(lmod1_original,no.engine_type)$Pr[2],3)
```

```
## [1] 0.831
```

```r
round(anova(lmod1_original,no.fuel_system)$Pr[2],3)
```

```
## [1] 0.15
```

```r
round(anova(lmod1_original,no.aspiration)$Pr[2],3)
```

```
## [1] 0.018
```

We can see an `NA` value in fuel_type. This indicates fuel_type is perfectly correlated with some combination of other covariates. We can see it from the F-test below.

```r
anova(lmod1_original,no.fuel_type)
```

```
## Analysis of Variance Table
##
## Model 1: highway_mpg ~ curb_weight + make + fuel_type + num_of_doors +
##     body_style + drive_wheels + num_of_cylinders + engine_type +
##     fuel_system + aspiration + stroke + compression_rate + bore +
##     peak_rpm + horsepower + normalized_losses
## Model 2: highway_mpg ~ curb_weight + make + num_of_doors + body_style +
##     drive_wheels + num_of_cylinders + engine_type + fuel_system +
##     aspiration + stroke + compression_rate + bore + peak_rpm +
##     horsepower + normalized_losses
##   Res.Df    RSS Df  Sum of Sq F Pr(>F)
## 1    144 646.72
## 2    144 646.72  0 2.2737e-13
```

We can see there is no difference in the model, so we exclude `fuel_type` from the model.

After deleting `fuel_type`, we just repeat what we did above, at each step we exclude the one categorical covarite which has the highest p-value from our regression model. The results are here, with all the steps and codes hidden.

```
##            stroke  compression_rate              bore           peak_rpm
##             0.918             0.002             0.086              0.172
##        horsepower normalized_losses       curb_weight
##             0.235             0.667             0.000
```

```
## [1] 0.029
## [1] 0.671
## [1] 0.864
## [1] 0.464
## [1] 0.007
## [1] 0.831
## [1] 0.034
## [1] 0.018
```

Remove `stroke`

```
##  compression_rate              bore          peak_rpm        horsepower
##             0.002             0.056             0.171             0.231
## normalized_losses       curb_weight
##             0.665             0.000
## [1] 0.019
## [1] 0.672
## [1] 0.863
## [1] 0.463
## [1] 0.006
## [1] 0.831
## [1] 0.033
## [1] 0.017
```

Remove `body_style`

```
##  compression_rate              bore          peak_rpm        horsepower
##             0.002             0.035             0.146             0.180
## normalized_losses       curb_weight
##             0.757             0.000
## [1] 0.016
## [1] 0.532
## [1] 0.371
## [1] 0.006
## [1] 0.79
## [1] 0.037
## [1] 0.015
```

Remove `engine_type`

```
##  compression_rate              bore          peak_rpm        horsepower
##             0.000             0.009             0.022             0.046
## normalized_losses       curb_weight
##             0.715             0.000
## [1] 0.001
```

```
## [1] 0.508

## [1] 0.303

## [1] 0

## [1] 0.009

## [1] 0.006
```

Remove `normalized_losses`

```
## compression_rate             bore         peak_rpm       horsepower
##            0.000            0.009            0.016            0.039
##      curb_weight
##            0.000

## [1] 0.001

## [1] 0.557

## [1] 0.163

## [1] 0

## [1] 0.009

## [1] 0.005
```

Remove `num_of_doors`

```
## compression_rate             bore         peak_rpm       horsepower
##            0.000            0.009            0.014            0.033
##      curb_weight
##            0.000

## [1] 0.001

## [1] 0.19

## [1] 0

## [1] 0.01

## [1] 0.005
```

Remove `drive_wheels`

```
## compression_rate             bore         peak_rpm       horsepower
##            0.000            0.005            0.006            0.044
##      curb_weight
##            0.000

## [1] 0.001

## [1] 0

## [1] 0.013

## [1] 0.005
```

```
cor(auto$horsepower, auto$bore)
```

```
## [1] 0.5769887
```

Since backward elimination tends overestimate the significance of estimators, and `horsepower` has its coefficient very close to 0.05, and `horsepower` has positive correlation with `bore`, we will remove it. If thereis any problem with the model, we can add it back.

remove `horsepower`

```
## compression_rate             bore         peak_rpm      curb_weight
##            0.001            0.035            0.038            0.000
```

```
## [1] 0.002
```

```
## [1] 0
```

```
## [1] 0.042
```

```
## [1] 0.041
```

We can see we still have every other coefficients been less than 0.05.

The order of removal of covariates are as follows: `stroke`; `body_style`; `engine_type`; `normalized_losses`; `num_of_doors`; `drive_wheels`; `horsepower`

The resulting model is as follows:

```
##                  Estimate Std. Error t value Pr(>|t|)
## aspirationturbo    -1.400      0.678  -2.064    0.041
## compression_rate    1.579      0.447   3.530    0.001
## bore               -2.695      1.267  -2.127    0.035
## peak_rpm           -0.001      0.001  -2.089    0.038
## curb_weight        -0.007      0.001  -6.892    0.000
```

```
## Analysis of Variance Table
##
## Response: highway_mpg
##                   Df Sum Sq Mean Sq F value    Pr(>F)
## make              20 4170.0 208.500 47.4634 < 2.2e-16 ***
## num_of_cylinders   5 1121.1 224.214 51.0405 < 2.2e-16 ***
## fuel_system        6 1714.3 285.722 65.0424 < 2.2e-16 ***
## aspiration         1  172.4 172.391 39.2435 3.396e-09 ***
## compression_rate   1  184.0 183.956 41.8761 1.163e-09 ***
## bore               1  227.0 226.991 51.6728 2.456e-11 ***
## peak_rpm           1    8.9   8.856  2.0161    0.1576
## curb_weight        1  208.6 208.631 47.4932 1.243e-10 ***
## Residuals        158  694.1   4.393
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Running F-test to see if the data favors the simpler model:

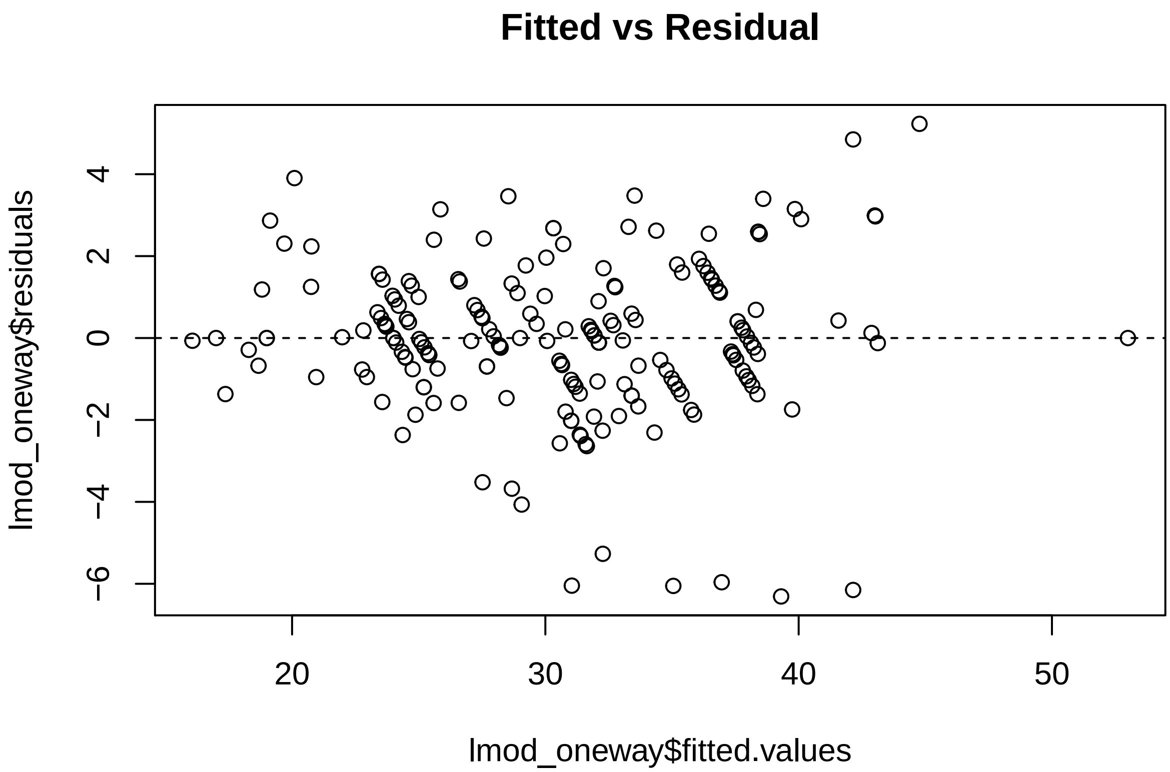
```
anova(lmod1_original,lmod_oneway)
```

```
## Analysis of Variance Table
##
## Model 1: highway_mpg ~ curb_weight + make + fuel_type + num_of_doors +
##     body_style + drive_wheels + num_of_cylinders + engine_type +
##     fuel_system + aspiration + stroke + compression_rate + bore +
##     peak_rpm + horsepower + normalized_losses
## Model 2: highway_mpg ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + bore + peak_rpm + curb_weight
##   Res.Df    RSS  Df Sum of Sq      F Pr(>F)
## 1    144 646.72
## 2    158 694.07 -14   -47.353 0.7531 0.7177
```

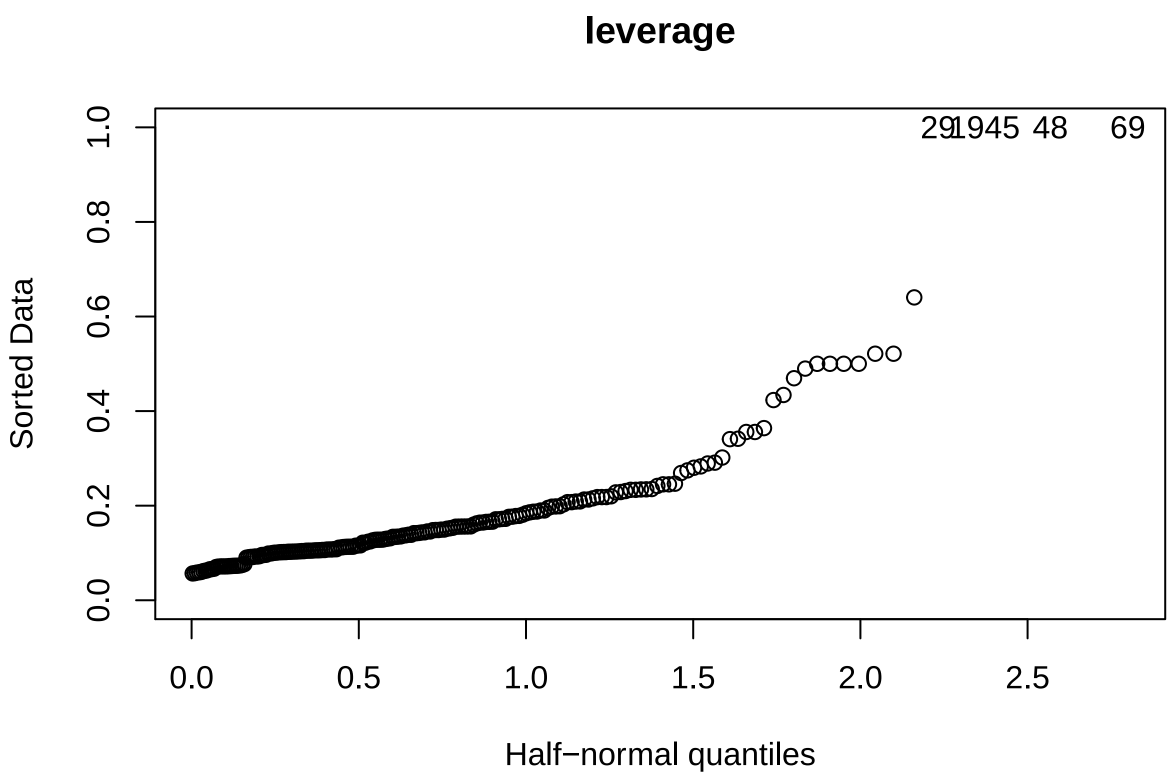
We can see that the p-value for F-test is much larger than 0.05, thus the data favors the simpler model.

Residual Plots

```
plot(lmod_oneway$fitted.values,lmod_oneway$residuals, main="Fitted vs Residual")
abline(h = 0, lty=2)
```

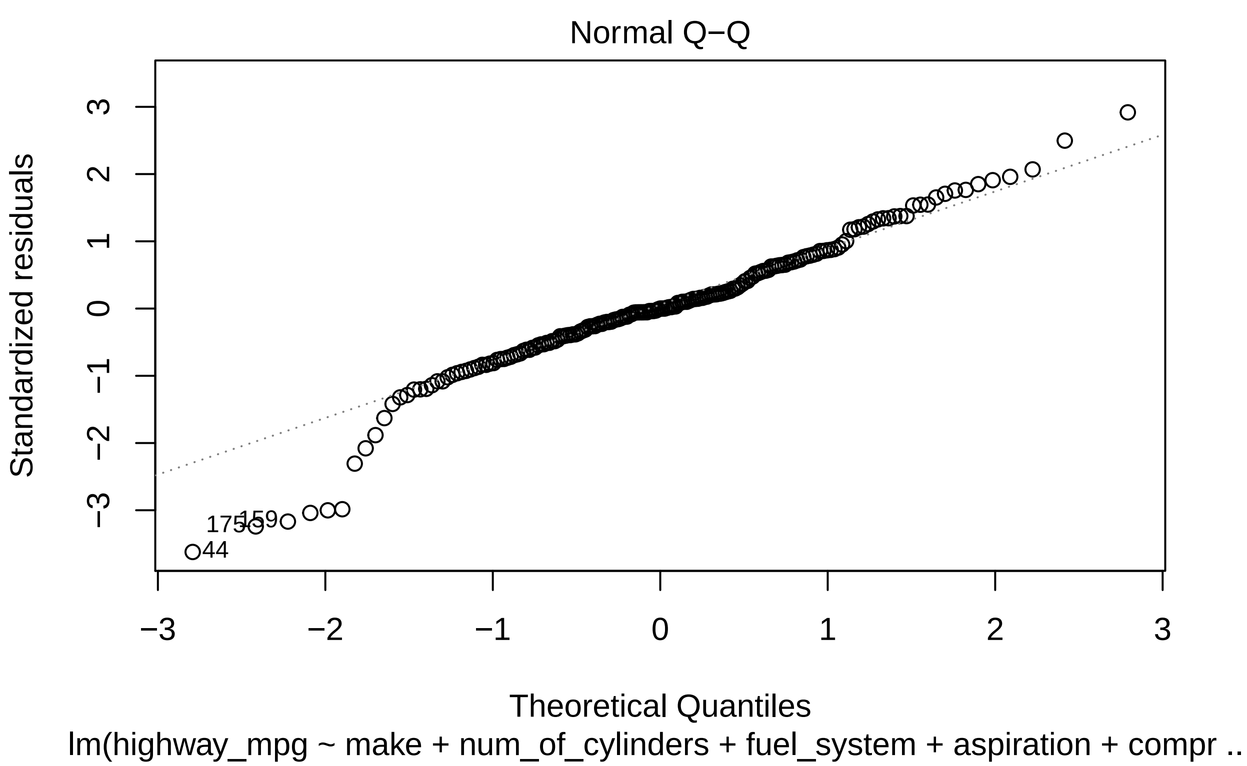



```
X <- model.matrix(lmod_oneway)
H <- hatvalues(lmod_oneway)
halfnorm(H, nlab=5, main="leverage")
```




```
plot(lmod_oneway, which =2)
```

```
## Warning: not plotting observations with leverage one:
##   19, 45, 48, 69
```



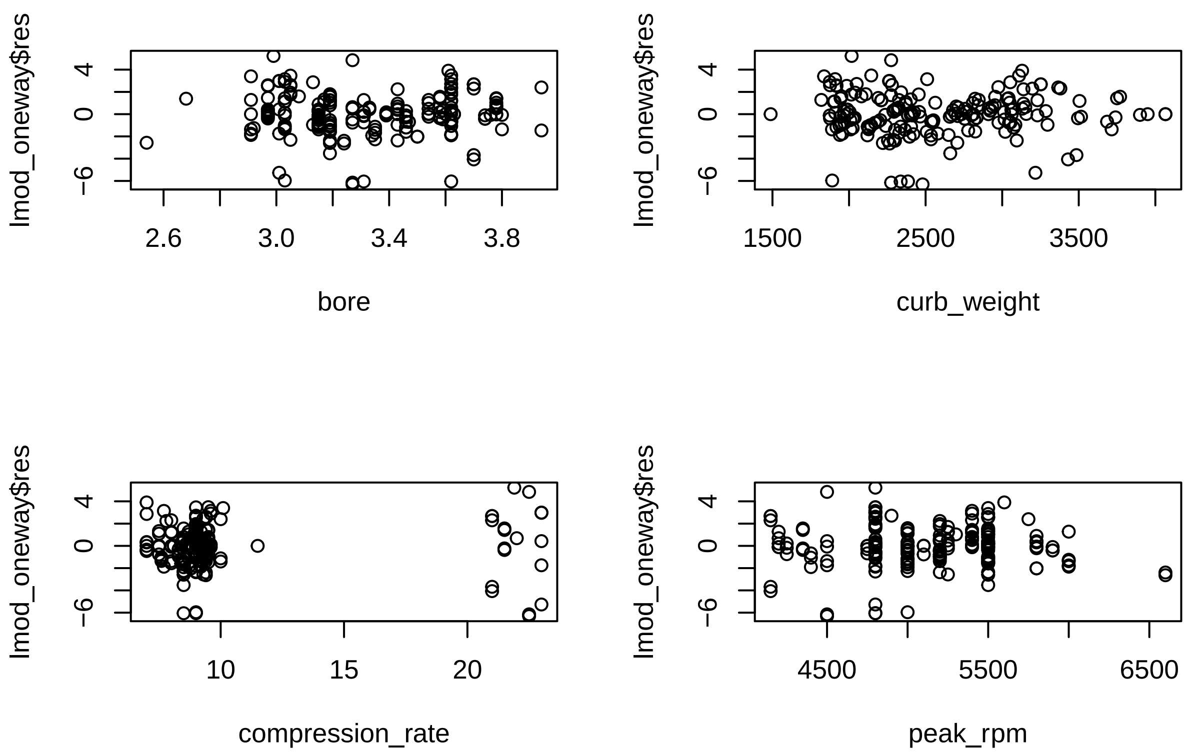
```
shapiro.test(lmod_oneway$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  lmod_oneway$residuals
## W = 0.95493, p-value = 7.518e-06
```

We see a similar residual plot as in the initial regression. This is a good indication because we did not introduce problems into the model by removing non-significant covariates. Note that the normality assumption still doesn't hold.

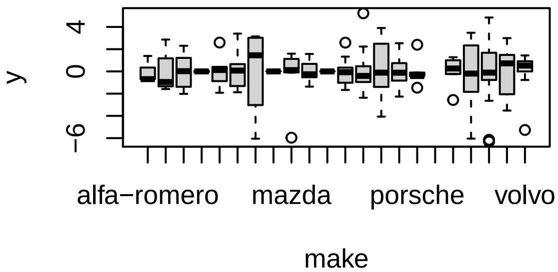
Let's now see the residual plots against continuous covariates.

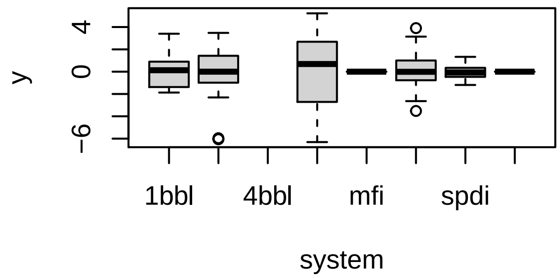
```
par(mfrow=c(2,2))
plot(as.numeric(auto$bore),lmod_oneway$res,xlab='bore')
plot(as.numeric(auto$curb_weight),lmod_oneway$res,xlab='curb_weight')
plot(as.numeric(auto$compression_rate),lmod_oneway$res,xlab='compression_rate')
plot(as.numeric(auto$peak_rpm),lmod_oneway$res,xlab='peak_rpm')
```

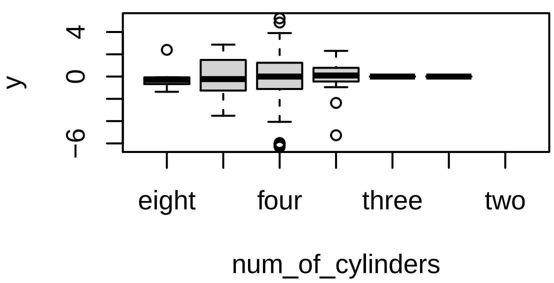


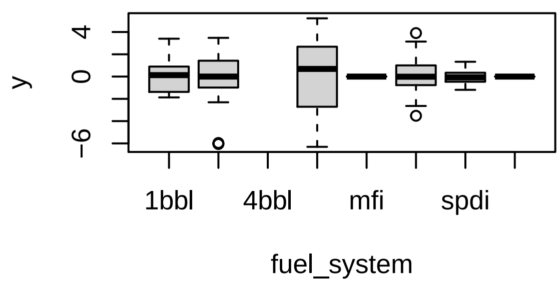
There is no clear red flag. We can see there are two clear clusters with `compression_rate`, maybe it is better to use this covarite as categorical but this is beyond the things that we should try. The variance looks constant across the two clusters, so we don't need to split the samples. For `bore` and `horsepower`, the non-linear trend is obvious, the variance is larger with smaller values, so we could consider transform it. Otherwise, the model's continuous covariates are okay.

```
par(mfrow=c(2,2))
plot(auto$make,lmod_oneway$res,xlab='make')
plot(auto$fuel_system,lmod_oneway$res,xlab='system')
plot(auto$num_of_cylinders,lmod_oneway$res,xlab='num_of_cylinders')
plot(auto$fuel_system,lmod_oneway$res,xlab='fuel_system')
```

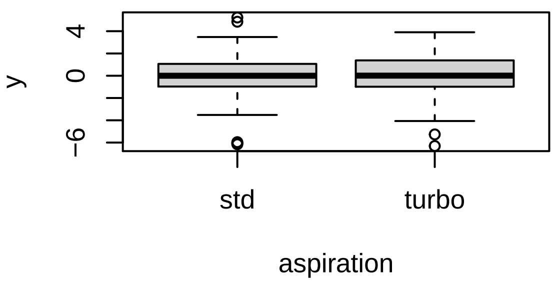









```
plot(auto$aspiration,lmod_oneway$res,xlab='aspiration')
```

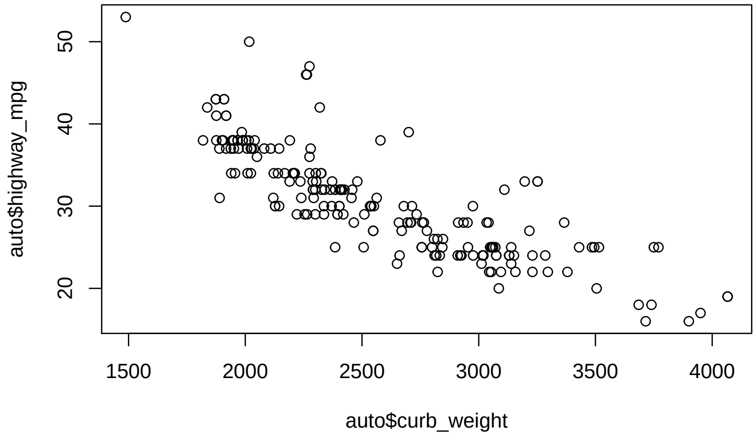


The variance looks constant across different levels.

6. Transformation

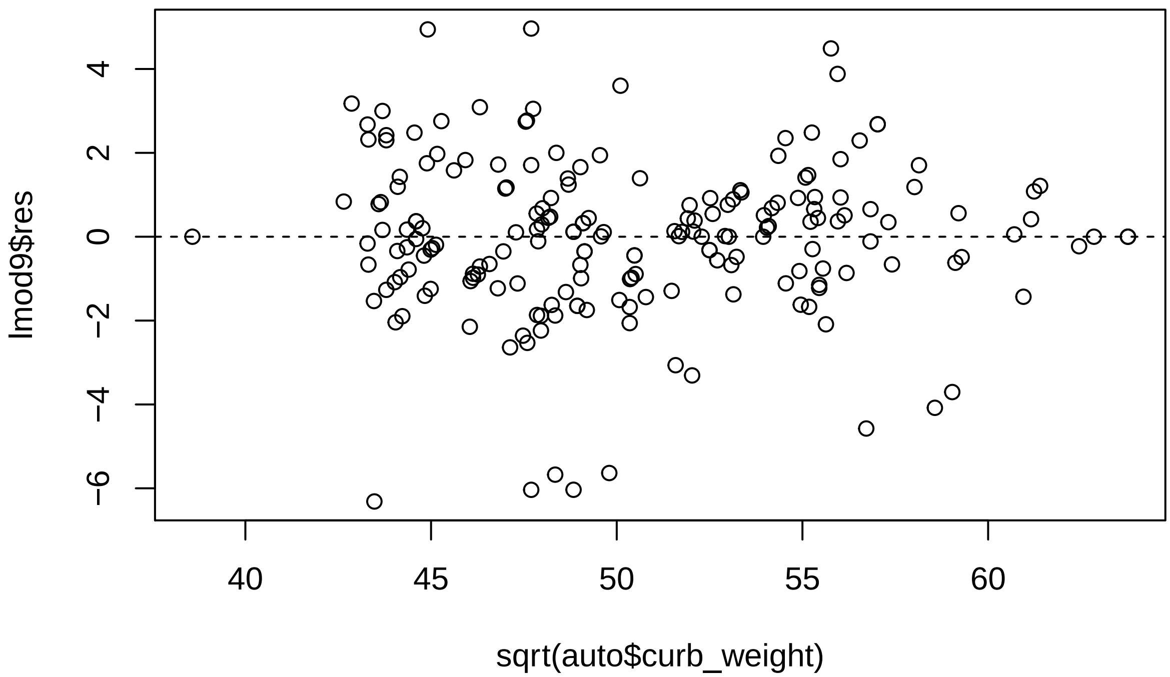
Transforming covariates

```
plot(auto$curb_weight, auto$highway_mpg)
```



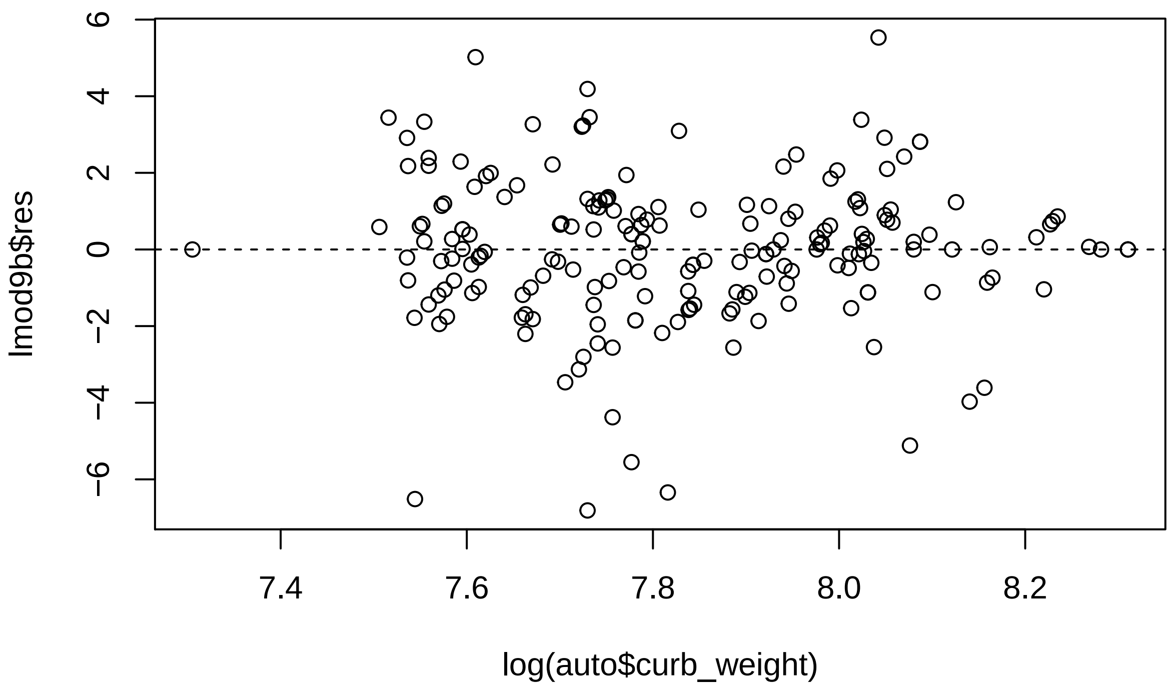
The trend is downward and we don't see obvious non-linearity. Because the `residual vs curb_weight` plot shows points agregated towards the center, we will first try square transformation, then log transformation.

```
lmod9 <- lm(formula = highway_mpg ~ make + num_of_cylinders + fuel_system +
    aspiration + compression_rate + bore + peak_rpm + horsepower +
    sqrt(curb_weight),
    data = auto)
plot(sqrt(auto$curb_weight), lmod9$res)
abline(h = 0, lty=2)
```




```
lmod9b <- lm(formula = highway_mpg ~ log(curb_weight) + compression_rate + make +
    fuel_type + num_of_cylinders + engine_type + aspiration,
    data = auto)
plot(log(auto$curb_weight), lmod9b$res)
```

```r
abline(h = 0, lty=2)
```



We can see curb_weight has its residual plot a lot better when using square root or log transformation. We will take log transformation as it is easier to interpreate.

```r
auto$curb_weight1 <- log(auto$curb_weight)
```

```r
lmod_oneway <- lm(highway_mpg~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+bore+
              peak_rpm+curb_weight1, auto)
round(summary(lmod_oneway)$coefficient[34:37,],3)
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## compression_rate     1.485      0.440   3.375    0.001
## bore                -2.286      1.250  -1.828    0.069
## peak_rpm            -0.001      0.001  -2.356    0.020
## curb_weight1       -18.612      2.495  -7.461    0.000
```

We can see that `bore` is now not statistically siginificant, we can consider exclude it from the model.

```r
lmod_oneway <- lm(highway_mpg~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1, auto)
round(summary(lmod_oneway)$coefficient[34:36,],3)
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## compression_rate     1.537      0.442   3.474    0.001
## peak_rpm            -0.001      0.001  -2.181    0.031
## curb_weight1       -21.232      2.057 -10.323    0.000
```

```r
anova(lmod_oneway)
```

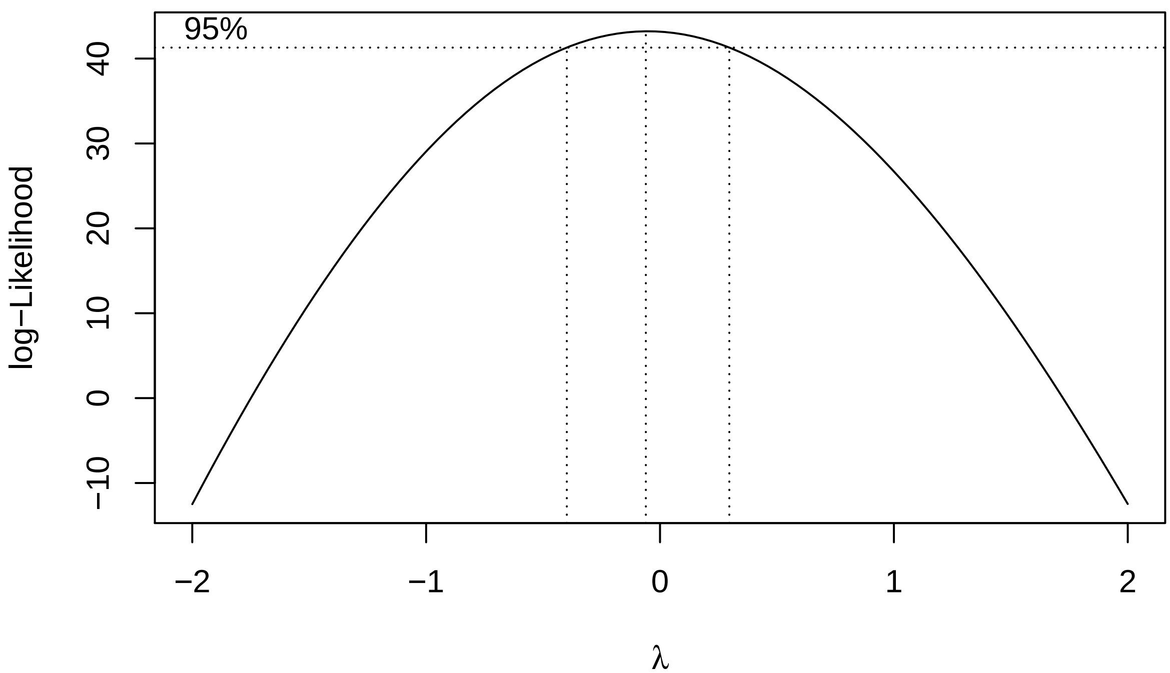
```
## Analysis of Variance Table
##
## Response: highway_mpg
```

```
##                    Df Sum Sq Mean Sq  F value    Pr(>F)    
## make               20 4170.0  208.50  48.6340 < 2.2e-16 ***
## num_of_cylinders    5 1121.1  224.21  52.2994 < 2.2e-16 ***
## fuel_system         6 1714.3  285.72  66.6466 < 2.2e-16 ***
## aspiration          1  172.4  172.39  40.2114 2.257e-09 ***
## compression_rate    1  184.0  183.96  42.9090 7.561e-10 ***
## peak_rpm            1    0.0    0.01   0.0024    0.9606    
## curb_weight1        1  456.9  456.89 106.5723 < 2.2e-16 ***
## Residuals         159  681.7    4.29                       
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Transforming the response

SInce the residual vs fitted plot also seems introducing non-linear trend, it would be helpful if we could transform highway_mpg(the response variable) into something else. We will use Box-Cox to choose a transformation.

```
boxcox(lmod_oneway)
```



We can see that 0 is not in 95% maximum log likelihood range, so we can use $\lambda = 0$, which is the same as taking log transformation with `highway_mpg`

```
lmod10<- lm(log(highway_mpg) ~ make + num_of_cylinders + fuel_system +
    aspiration + compression_rate + peak_rpm + curb_weight1,
    data = auto)
round(summary(lmod10)$coefficient[34:36,],3)
```

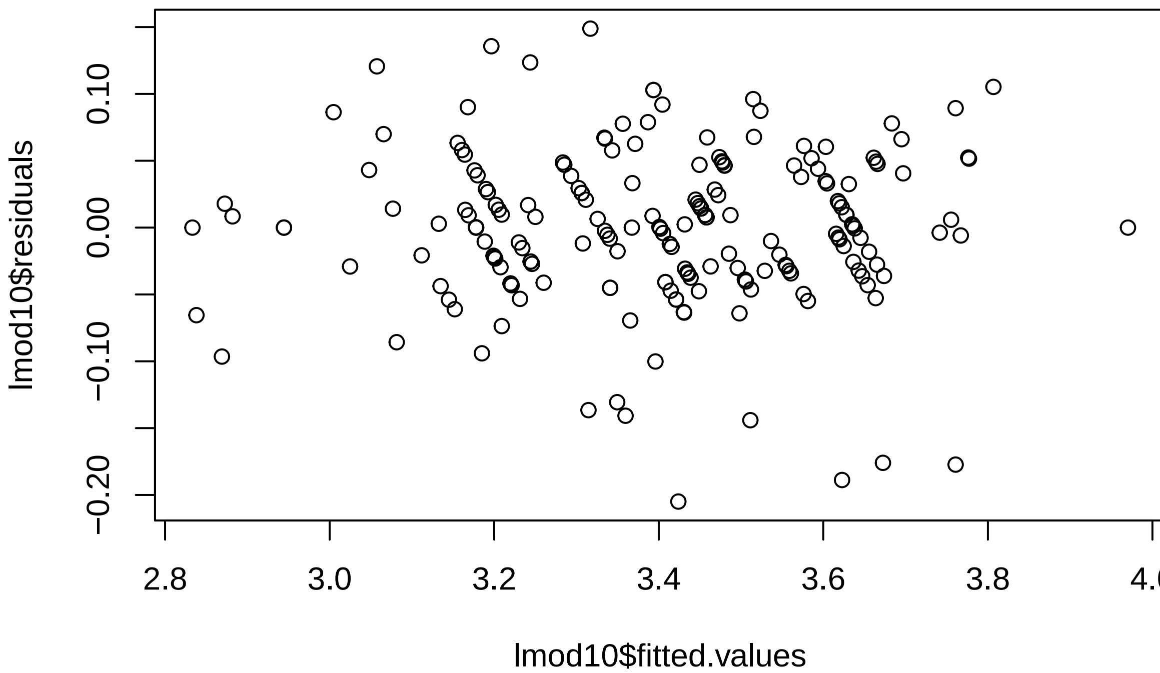
```
##                   Estimate Std. Error t value Pr(>|t|)
## compression_rate     0.064      0.014   4.735    0.000
## peak_rpm             0.000      0.000  -2.724    0.007
## curb_weight1        -0.637      0.063 -10.097    0.000
```

```
anova(lmod10)
```

```
## Analysis of Variance Table
```

```
##
## Response: log(highway_mpg)
##                    Df Sum Sq Mean Sq  F value    Pr(>F)
## make               20 4.8373 0.24186  59.8774 < 2.2e-16 ***
## num_of_cylinders    5 1.3622 0.27244  67.4457 < 2.2e-16 ***
## fuel_system         6 1.6577 0.27628  68.3980 < 2.2e-16 ***
## aspiration          1 0.1677 0.16772  41.5227 1.324e-09 ***
## compression_rate    1 0.2390 0.23900  59.1678 1.426e-12 ***
## peak_rpm            1 0.0013 0.00125   0.3103    0.5783
## curb_weight1        1 0.4118 0.41182 101.9537 < 2.2e-16 ***
## Residuals         159 0.6423 0.00404
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(lmod10$fitted.values, lmod10$residuals)
```



We can see we actually improved our coefficients as their p-value gets smaller, the residual plot looks better and we have a higher adjusted R-square. We will use the log transformation of `highway_mpg`

```
auto$highway_mpg1 <- log(auto$highway_mpg)
```

7. Add Interaction Term

Again our model is the following:

```
lmod_oneway <- lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1, auto)
```

Start with untransformed data. We have 7 terms, so we need to watch out for multiple testing issue. In this case we will use Bonferoni correction. We should be careful that we should only add one interaction term at a time due to the nature of F-test. If there are several term that meets the requirement, we will choose the one with the least p-value.

```
0.05/(3*7)
```

```
## [1] 0.002380952
```

```
anova(lmod_oneway,lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1
              +make:compression_rate
              , auto))
```

```
## Analysis of Variance Table
##
## Model 1: highway_mpg1 ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + peak_rpm + curb_weight1
## Model 2: highway_mpg1 ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + peak_rpm + curb_weight1 + make:compression_rate
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1    159 0.64225
## 2    143 0.46688 16   0.17538 3.3572 5.189e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

After checking all two-way terms, the least p-value for F test we found was adding the interaction term `make:peak_rpm`.

Add interaction term make:peak_rpm

```
lmod_twoway1 <- lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1
              +make:compression_rate
             , auto)
```

At each step we just repeat the same thing until every possible interaction term has p-value greater than the Bonferoni correction. But we need to adjust the Boferoni correction at each step.

```
0.05/(3*7 - 1)
```

```
## [1] 0.0025
```

```
anova(lmod_twoway1,lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1
              +make:compression_rate + make:fuel_system
              ,auto))
```

```
## Analysis of Variance Table
```

```
##
## Model 1: highway_mpg1 ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + peak_rpm + curb_weight1 + make:compression_rate
## Model 2: highway_mpg1 ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + peak_rpm + curb_weight1 + make:compression_rate +
##     make:fuel_system
##   Res.Df     RSS Df Sum of Sq      F   Pr(>F)
## 1    143 0.46688
## 2    132 0.37426 11  0.092621 2.9698 0.001497 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

After checking all two-way terms, the least p-value for F test we found was adding the interaction term `make:fuel_system`.

Add interaction term make:fuel_system

```
lmod_twoway2 <- lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1
              +make:compression_rate + make:fuel_system
              ,auto)
```

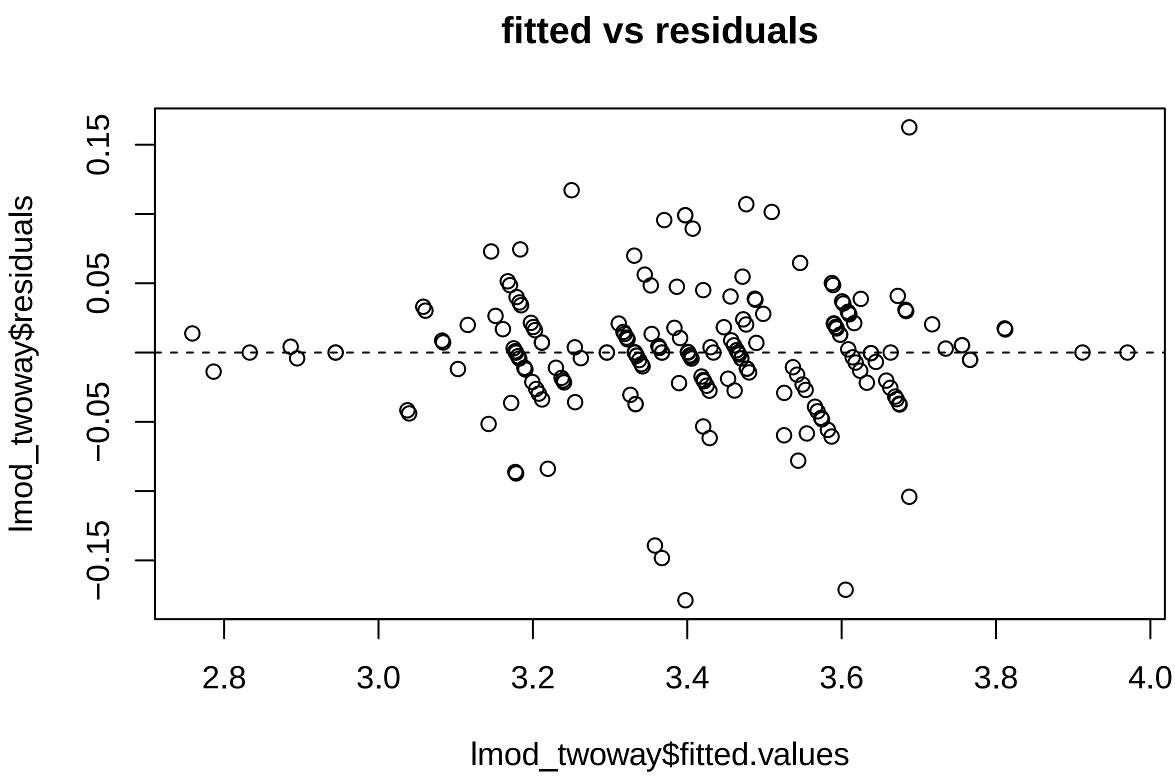
After checking all the other interaction terms, we find no other interaction term will improve the model in a significant way. Lets check how our model compares with the oneway-only model.

```
lmod_twoway <- lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1
              +make:compression_rate + make:fuel_system
              ,auto)

anova(lmod_oneway, lmod_twoway)
```

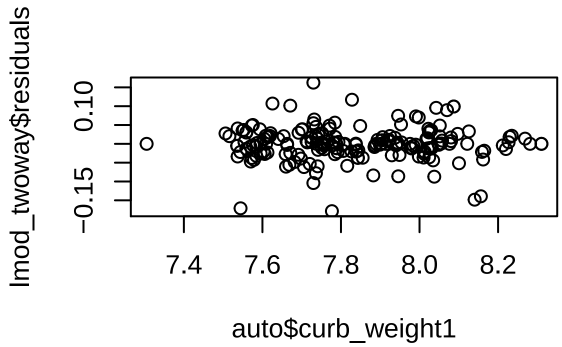
```
## Analysis of Variance Table
##
## Model 1: highway_mpg1 ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + peak_rpm + curb_weight1
## Model 2: highway_mpg1 ~ make + num_of_cylinders + fuel_system + aspiration +
##     compression_rate + peak_rpm + curb_weight1 + make:compression_rate +
##     make:fuel_system
##   Res.Df     RSS Df Sum of Sq      F   Pr(>F)
## 1    159 0.64225
## 2    132 0.37426 27     0.268 3.5008 8.62e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

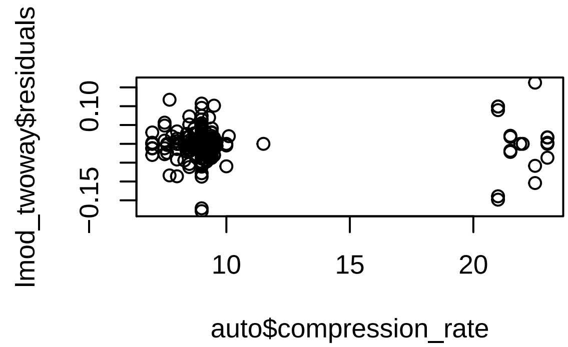
It is good that after adding two interaction terms, the model is significantly better than one way model, suggesting we should keep it. Let's check the residual plot.

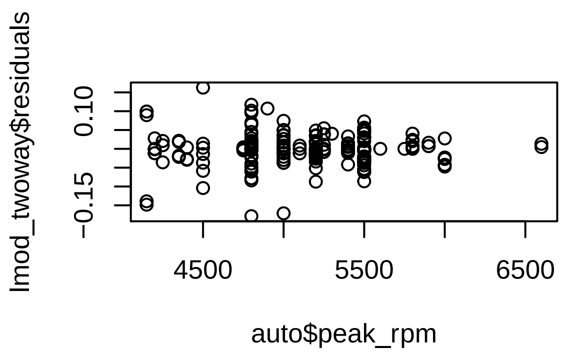
```
plot(lmod_twoway$fitted.values, lmod_twoway$residuals, main="fitted vs residuals")
abline(h = 0, lty=2)
```

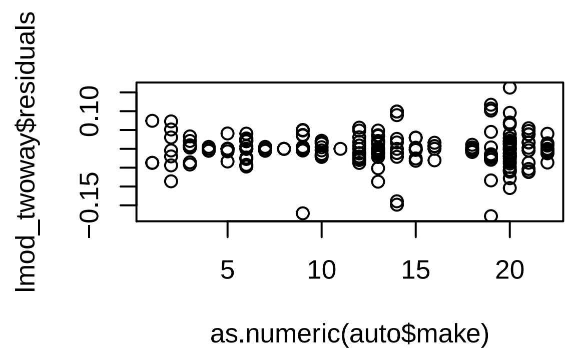



```
par(mfrow=c(2,2))
plot(auto$curb_weight1, lmod_twoway$residuals)
plot(auto$compression_rate, lmod_twoway$residuals)
plot(auto$peak_rpm, lmod_twoway$residuals)
plot(as.numeric(auto$make), lmod_twoway$residuals)
```

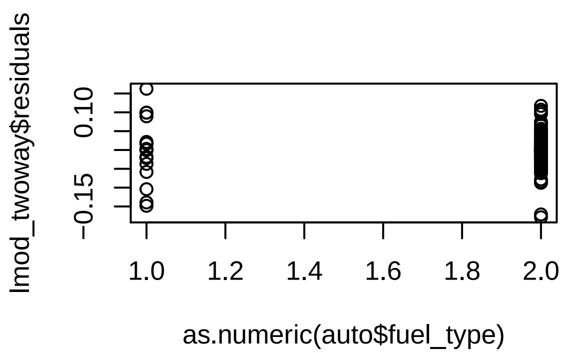


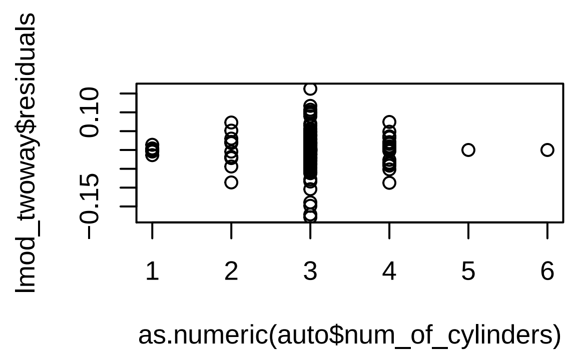


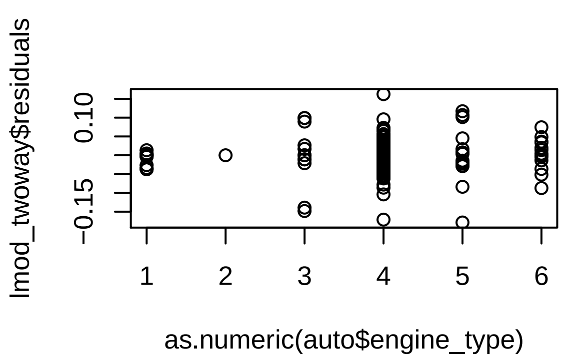


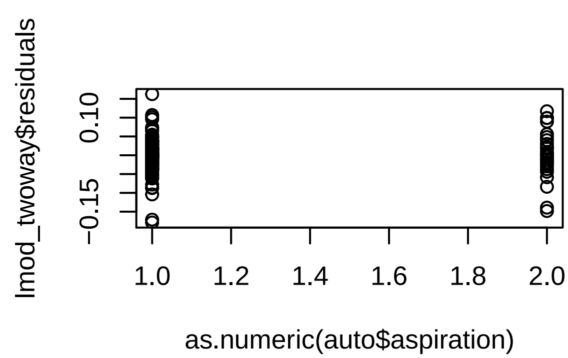



```
plot(as.numeric(auto$fuel_type), lmod_twoway$residuals)
plot(as.numeric(auto$num_of_cylinders), lmod_twoway$residuals)
plot(as.numeric(auto$engine_type), lmod_twoway$residuals)
plot(as.numeric(auto$aspiration), lmod_twoway$residuals)
```









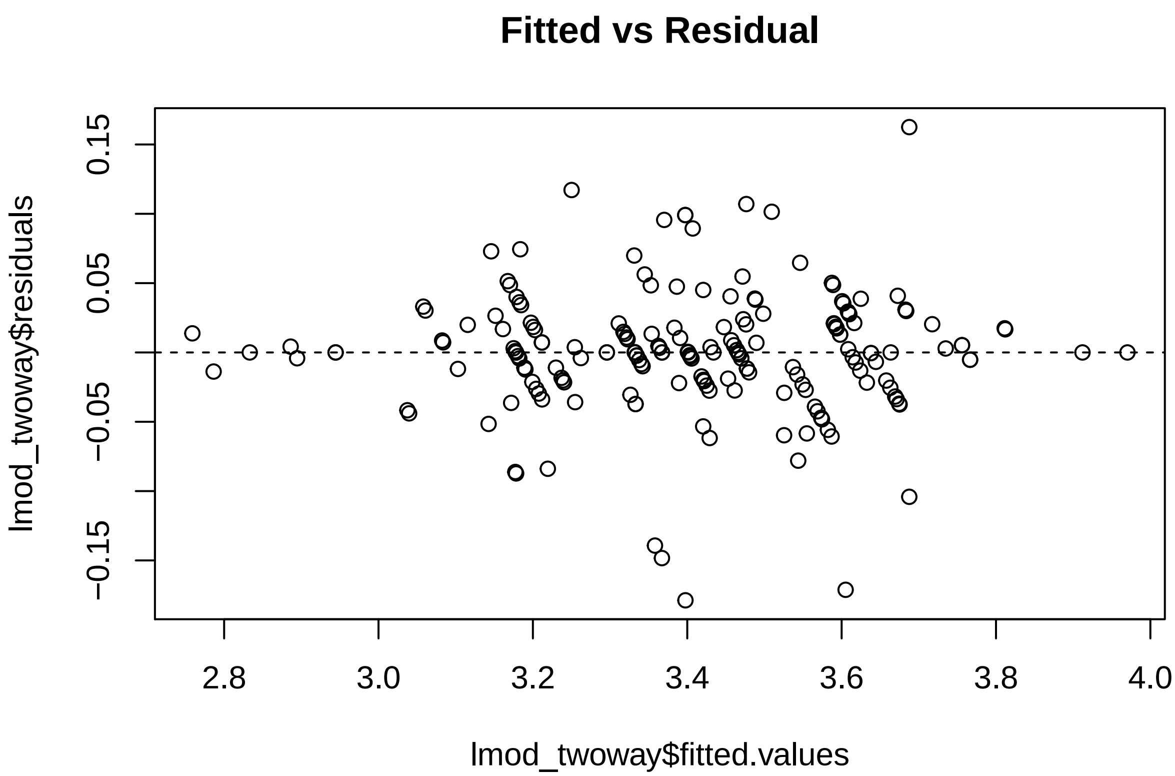
We can see the residual plots are improved after adding the two interaction terms, suggesting we should keep it. There is no clear indication of appearance of non-constant variance. Also, we can see the number of positive residuals and negative residuals are much even out than before.

8. Final Model

```
# Here is our final model
# Again  note that highway_mpg1 = log(highway_mpg) and curb_weight1 = log(curb_weight)
lmod_twoway <- lm(highway_mpg1~make+num_of_cylinders+
              fuel_system+aspiration+compression_rate+
              peak_rpm+curb_weight1
              +make:compression_rate + make:fuel_system
              ,auto)
```

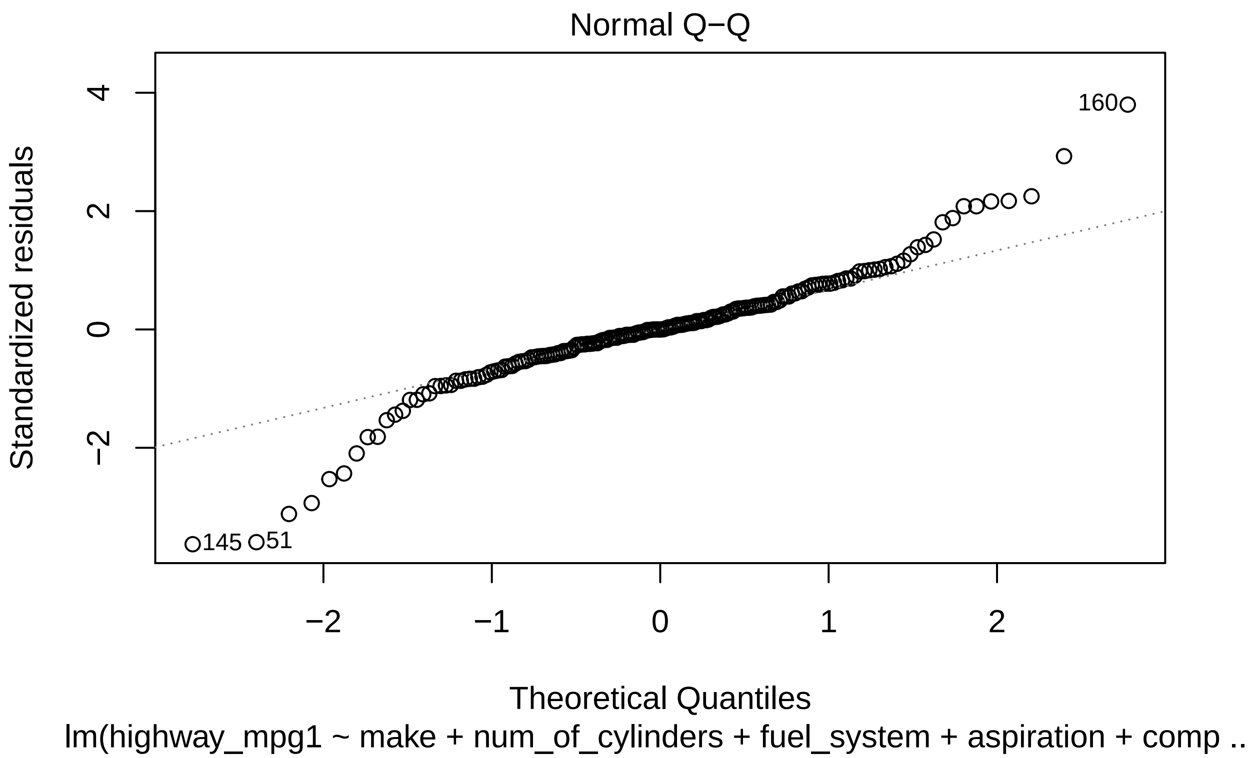
Fitted value vs Residuals

```
plot(lmod_twoway$fitted.values, lmod_twoway$residuals, main="Fitted vs Residual")
abline(h = 0, lty=2)
```



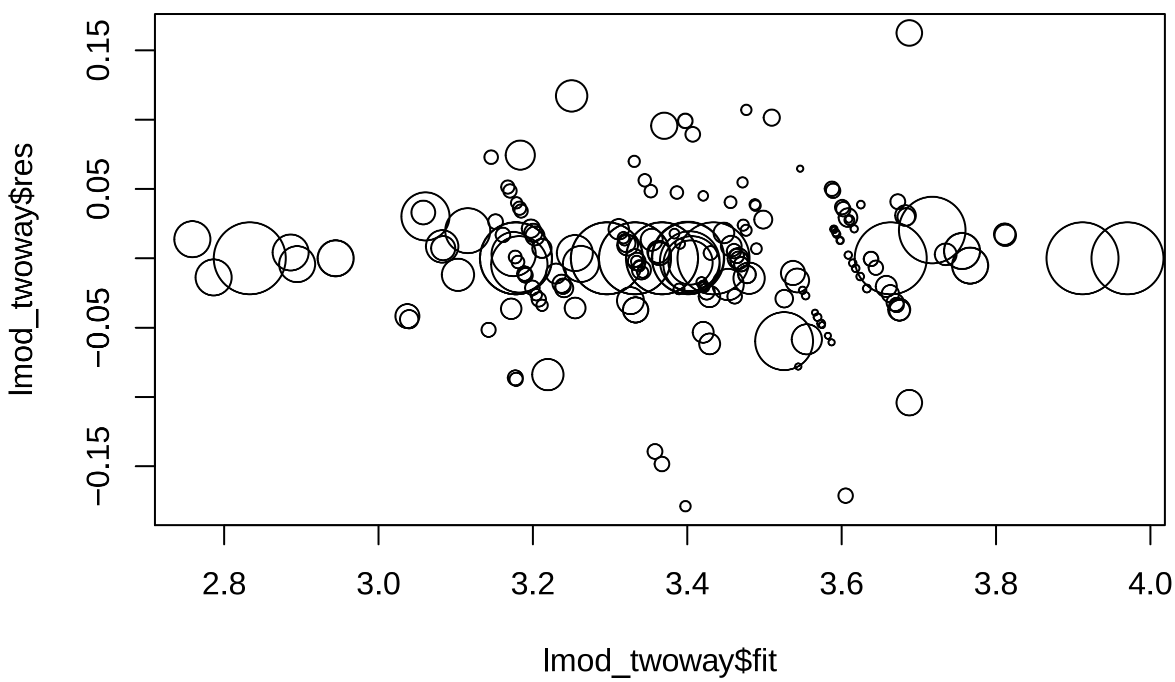

```
plot(lmod_twoway, which=2)
```

```
## Warning: not plotting observations with leverage one:
##   19, 40, 41, 42, 45, 48, 59, 60, 69, 84, 111, 123, 194
```



We are seeing many points form many "tilted lines". This is due to the nature of the data, where we have lots of different levels in each categorical variable, and the data we have is about cars, so it is likely those cars with the same make that form that phenomenon.

```
plot(lmod_twoway$fit,lmod_twoway$res,cex=5*hatvalues(lmod_twoway))
```



There is still some points carrying large residuals, but their leverage is not high, and we do not want to over-fit the model. The variance is better than the original model, so we will keep this.

Discussion

Last but not least, we should interpret this model.

```
summary(lmod_twoway)$coefficients
```

```
##                                        Estimate   Std. Error      t value
## (Intercept)                        1.148860e+01 2.532337e+00  4.536759277
## makeaudi                           8.538399e-01 3.451040e-01  2.474152624
## makebmw                            2.121307e-01 4.925733e-01  0.430658213
## makechevrolet                      1.705347e-01 5.636341e-02  3.025626774
## makedodge                         -3.725675e-01 5.591407e-01 -0.666321613
## makehonda                         -6.377068e-02 5.696470e-01 -0.111947716
## makeisuzu                         -1.860768e+00 6.054053e-01 -3.073590060
## makejaguar                         1.824043e-02 6.117349e-02  0.298175443
## makemazda                          3.251466e-01 7.934452e-01  0.409790909
## makemercedes-benz                 -6.851588e+00 3.869367e+00 -1.770725832
## makemercury                        7.966058e-02 6.750180e-02  1.180125225
## makemitsubishi                    -6.539928e-01 4.332299e-01 -1.509574542
## makenissan                         9.585992e-01 3.641768e-01  2.632236130
## makepeugot                         1.390806e+00 4.108562e-01  3.385140701
## makeplymouth                      -8.089279e-01 6.194461e-01 -1.305889234
## makeporsche                        7.301148e+01 4.563677e+01  1.599839053
## makesaab                           2.347961e+00 1.566576e+00  1.498784689
## makesubaru                        -1.375241e-01 3.531432e-01 -0.389428658
## maketoyota                         5.289144e-01 6.507006e-01  0.812838284
## makevolkswagen                     7.394086e-02 4.389617e-01  0.168444894
## makevolvo                          7.500862e-02 3.984249e-02  1.882628818
## num_of_cylindersfive              -4.079141e+00 2.401435e+00 -1.698626386
## num_of_cylindersfour              -3.935065e+00 2.401723e+00 -1.638433887
## num_of_cylinderssix               -4.026314e+00 2.402506e+00 -1.675880641
## num_of_cylindersthree             -3.875458e+00 2.403412e+00 -1.612481731
## num_of_cylinderstwelve            -4.393288e+00 2.393769e+00 -1.835301716
## fuel_system2bbl                   -3.081869e-02 6.112759e-02 -0.504169877
## fuel_systemidi                    -9.783762e-01 3.515136e-01 -2.783323603
## fuel_systemmfi                    -1.124617e-01 1.036231e-01 -1.085296201
## fuel_systemmpfi                   -1.020394e-01 5.789198e-02 -1.762583004
## fuel_systemspdi                    1.468323e-01 1.423571e-01  1.031436462
## fuel_systemspfi                   -1.288327e-01 8.954501e-02 -1.438748207
## aspirationturbo                   -8.377573e-02 2.322816e-02 -3.606644509
## compression_rate                   7.552913e-02 2.377971e-02  3.176201229
## peak_rpm                          -6.987328e-05 2.142271e-05 -3.261644884
## curb_weight1                      -5.672327e-01 8.109225e-02 -6.994907127
## makeaudi:compression_rate         -8.709655e-02 3.719061e-02 -2.341896521
## makebmw:compression_rate          -1.652103e-02 5.634361e-02 -0.293219248
## makedodge:compression_rate         7.407800e-02 7.261106e-02  1.020202731
## makehonda:compression_rate         1.111523e-02 6.255365e-02  0.177691090
## makeisuzu:compression_rate         2.115940e-01 6.570395e-02  3.220415040
## makemazda:compression_rate        -3.253788e-02 9.781969e-02 -0.332631213
## makemercedes-benz:compression_rate  3.280402e-01 1.796899e-01  1.825591019
## makemitsubishi:compression_rate    8.210423e-02 5.130601e-02  1.600284939
## makenissan:compression_rate       -9.975797e-02 4.064013e-02 -2.454666382
## makepeugot:compression_rate       -1.646218e-01 4.892351e-02 -3.364880169
## makeplymouth:compression_rate      9.914416e-02 6.734215e-02  1.472245177
## makeporsche:compression_rate      -7.681259e+00 4.803514e+00 -1.599091856
```

```
## makesaab:compression_rate          -2.494485e-01 1.697212e-01 -1.469755130
## makesubaru:compression_rate         2.780466e-02 4.105941e-02  0.677181155
## maketoyota:compression_rate        -5.155035e-02 7.107921e-02 -0.725252121
## makevolkswagen:compression_rate     1.977095e-03 4.861379e-02  0.040669431
## makedodge:fuel_system2bbl          -2.257615e-01 1.342340e-01 -1.681849902
## makehonda:fuel_system2bbl          -4.009286e-02 7.906711e-02 -0.507073865
## makemazda:fuel_system2bbl          -8.671237e-04 9.447411e-02 -0.009178426
## makemitsubishi:fuel_system2bbl      4.279575e-03 1.424800e-01  0.030036329
## makenissan:fuel_system2bbl          2.155133e-02 4.312141e-02  0.499782684
## makesubaru:fuel_system2bbl         -1.708333e-01 4.791557e-02 -3.565297369
## makemazda:fuel_systemidi            5.926233e-01 1.377639e+00  0.430173143
## makenissan:fuel_systemidi           1.560547e+00 5.495089e-01  2.839893843
## makepeugot:fuel_systemidi           2.263369e+00 6.407462e-01  3.532394802
## maketoyota:fuel_systemidi           7.430198e-01 9.627680e-01  0.771753813
## makevolkswagen:fuel_systemidi       9.668588e-02 6.857877e-01  0.140985161
##                                         Pr(>|t|)
## (Intercept)                        1.269678e-05
## makeaudi                           1.462412e-02
## makebmw                            6.674193e-01
## makechevrolet                      2.983293e-03
## makedodge                          5.063684e-01
## makehonda                          9.110349e-01
## makeisuzu                          2.570221e-03
## makejaguar                         7.660381e-01
## makemazda                          6.826236e-01
## makemercedes-benz                  7.891449e-02
## makemercury                        2.400728e-01
## makemitsubishi                     1.335420e-01
## makenissan                         9.493620e-03
## makepeugot                         9.372705e-04
## makeplymouth                       1.938615e-01
## makeporsche                        1.120251e-01
## makesaab                           1.363173e-01
## makesubaru                         6.975866e-01
## maketoyota                         4.177744e-01
## makevolkswagen                     8.664911e-01
## makevolvo                          6.195172e-02
## num_of_cylindersfive               9.174546e-02
## num_of_cylindersfour               1.037127e-01
## num_of_cylinderssix                9.612868e-02
## num_of_cylindersthree              1.092455e-01
## num_of_cylinderstwelve             6.871194e-02
## fuel_system2bbl                    6.149825e-01
## fuel_systemidi                     6.170729e-03
## fuel_systemmfi                     2.797686e-01
## fuel_systemmpfi                    8.028530e-02
## fuel_systemspdi                    3.042223e-01
## fuel_systemspfi                    1.525888e-01
## aspirationturbo                    4.387148e-04
## compression_rate                   1.857982e-03
## peak_rpm                           1.409800e-03
## curb_weight1                       1.199050e-10
## makeaudi:compression_rate          2.068243e-02
## makebmw:compression_rate           7.698150e-01
```

```
## makedodge:compression_rate         3.094985e-01
## makehonda:compression_rate         8.592380e-01
## makeisuzu:compression_rate         1.611729e-03
## makemazda:compression_rate         7.399404e-01
## makemercedes-benz:compression_rate 7.017226e-02
## makemitsubishi:compression_rate    1.119261e-01
## makenissan:compression_rate        1.540349e-02
## makepeugot:compression_rate        1.002939e-03
## makeplymouth:compression_rate      1.433350e-01
## makeporsche:compression_rate       1.121911e-01
## makesaab:compression_rate          1.440075e-01
## makesubaru:compression_rate        4.994761e-01
## maketoyota:compression_rate        4.695811e-01
## makevolkswagen:compression_rate    9.676209e-01
## makedodge:fuel_system2bbl          9.496228e-02
## makehonda:fuel_system2bbl          6.129491e-01
## makemazda:fuel_system2bbl          9.926906e-01
## makemitsubishi:fuel_system2bbl     9.760834e-01
## makenissan:fuel_system2bbl         6.180602e-01
## makesubaru:fuel_system2bbl         5.068120e-04
## makemazda:fuel_systemidi           6.677712e-01
## makenissan:fuel_systemidi          5.228195e-03
## makepeugot:fuel_systemidi          5.679981e-04
## maketoyota:fuel_systemidi          4.416397e-01
## makevolkswagen:fuel_systemidi      8.880966e-01
```

There are three continuous covariates: `peak_rpm`, `compression_rate` and `curb_weight`.

Compression rate, or compression ratio, when higher, means higher combustion efficiency (from wikipedia). It makes sense that it is positively related to miles per gallon as shown by our model.

As we all know, an engine doesn't necessarily produce its best power at the peak rpm. Therefore, the peak rpm being large may suggest the inefficiency of energy usage, leading to fewer miles per gallon. This fact is also reflected by our model by a negative coefficient for covariate `peak_rpm`.

The higher the curb weight is, the more energy a car consumes. It is also reflected in our model that `curb_weight` has negative relationship with the miles per gallon values. The transformation of this term may indicate the non-linear relationship between energy utility and the weight of a car, which further reflects in the transformed `highway_mpg`.

The categorical covariates included in the final model are manufacturers, number of cylinders, fuel systems, and aspiration. These are related to the hardware of a car's engine. It is easily understandable that different manufacturers produce different models, causing different mpg when driving. The fuel systems, aspiration, and number of cylinders are really hard to explain because these are the engine design and they should have effects on miles per gallon certainly. One thing we want to mention here is that we do not consider number of cylinders as a numerical covariate, but rather a categorical covariate because the numbers of cylinders in an engine are just design aspect of the engine, and a larger number does not necessarily represent better engine, or higher energy efficiency. So we treat it as categorical, to make the model more natural.

All the other non-significant covariates and terms removed by us can be explained by two possible reasons: (1) essentially they do not have anything to do with miles per gallon, like the number of doors. We cannot count on these specifications to infer how good a car will behave on highway. (2) the non-significance can be explained by the connections of these covariates, and the connections are underlying the nature of vehicles. For example, a car with larger length and height is very likely to have larger weight. So the model only include one most significant term, to show the most important covariate.

The interaction terms kept here in the final model are from computation, and they can be interpreted directly by what they are. Generally speaking, even with the same fuel system built, different engines from different manufactures will definitely have different performance in energy usage. That's why `make` has two two-way interaction terms added into our model. Some makers make engines better than others, and these effects are shown with the coefficients.

Since we are not expert in car engines, we cannot judge how good the model is realistically. But in the end, we can see our residual plots for fitted value looks good, in a sense that there is little indication of non-constant variance or non-linearity. Each categorical vs residual plots are showing reasonable spread of variances, meaning there is no indication of non-constant residuals. Finally, each continuous variable has reasonable residual plot as well.