cars_analysis_final.R

rivka

2024-12-22

```
#loading cars data and summarizing
# original data:
cars2 <- read.csv("C:/Users/rivka/Documents/college/intro to data science/final_project/auto-mpg(1).csv
summary(cars2)
```

```
##       mpg           cylinder      displacement    horsepower
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Length:398
##  1st Qu.:17.50   1st Qu.:4.000   1st Qu.:104.2   Class :character
##  Median :23.00   Median :4.000   Median :148.5   Mode  :character
##  Mean   :23.51   Mean   :5.455   Mean   :193.4
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:262.0
##  Max.   :46.60   Max.   :8.000   Max.   :455.0
##      weight      acceleration     model.year        origin
##  Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
##  1st Qu.:2224   1st Qu.:13.82   1st Qu.:73.00   1st Qu.:1.000
##  Median :2804   Median :15.50   Median :76.00   Median :1.000
##  Mean   :2970   Mean   :15.57   Mean   :76.01   Mean   :1.573
##  3rd Qu.:3608   3rd Qu.:17.18   3rd Qu.:79.00   3rd Qu.:2.000
##  Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000
##    car.name
##  Length:398
##  Class :character
##  Mode  :character
##
##
##
```

```
### (diff tools i used in analyzing the data)
head(sort(cars2$car.name))
```

```
## [1] "amc ambassador brougham" "amc ambassador dpl"
## [3] "amc ambassador sst"      "amc concord"
## [5] "amc concord"             "amc concord d/l"
```

```
table(is.na(cars2$displacement))
```

```
##
## FALSE
##   398
```

```r
#checking data type of data
str(cars2)
```

```
## 'data.frame':    398 obs. of  9 variables:
##  $ mpg         : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cylinder    : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ displacement: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ horsepower  : chr  "130" "165" "150" "150" ...
##  $ weight      : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
##  $ acceleration: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ model.year  : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origin      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ car.name    : chr  "chevrolet chevelle malibu" "buick skylark 320" "plymouth satellite" "amc rebel
```

```r
### i decided to convert the column car name to car company since there is not enough data to analyze by
### looking at the car name column, i see that the car company is the first word in the name
# so i need to extract the first word from each car name to determine the car company
cars2$car_company <- sapply(strsplit(as.character(cars2$car.name), " "), `[`, 1)

# convert the car_company column to a factor
cars2$car_company <- as.factor(cars2$car_company)

# view the resulting dataset with the new car company column
head(cars2)
```

```
##   mpg cylinder displacement horsepower weight acceleration model.year origin
## 1  18        8          307        130   3504         12.0         70      1
## 2  15        8          350        165   3693         11.5         70      1
## 3  18        8          318        150   3436         11.0         70      1
## 4  16        8          304        150   3433         12.0         70      1
## 5  17        8          302        140   3449         10.5         70      1
## 6  15        8          429        198   4341         10.0         70      1
##                   car.name car_company
## 1 chevrolet chevelle malibu   chevrolet
## 2         buick skylark 320       buick
## 3        plymouth satellite    plymouth
## 4             amc rebel sst         amc
## 5               ford torino        ford
## 6          ford galaxie 500        ford
```

```r
sort(unique(cars2$car_company))
```

```
##  [1] amc           audi          bmw           buick         cadillac
##  [6] capri         chevroelt     chevrolet     chevy         chrysler
## [11] datsun        dodge         fiat          ford          hi
## [16] honda         maxda         mazda         mercedes      mercedes-benz
## [21] mercury       nissan        oldsmobile    opel          peugeot
## [26] plymouth      pontiac       renault       saab          subaru
## [31] toyota        toyouta       triumph       vokswagen     volkswagen
## [36] volvo         vw
## 37 Levels: amc audi bmw buick cadillac capri chevroelt chevrolet ... vw
```

```r
#delete car.name column because no longer needed
cars2 <- subset(cars2, select = -car.name)

### i realized there are spelling mistakes and abbreviations to some car company categories,
# so combined the categories as appropriate
library(dplyr) #we will need to use "recode" from the library "dplyr"
```

```
## Warning: package 'dplyr' was built under R version 4.4.2
```

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

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

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

```r
cars2$car_company <- recode(cars2$car_company,
                            "chevroelt" = "chevrolet",
                            "chevy" = "chevrolet",
                            "mercedes-benz" = "mercedes",
                            "maxda" = "mazda",
                            "toyouta" = "toyota",
                            "vokswagen" = "volkswagen",
                            "vw" = "volkswagen")


# checking unique levels after correction
sort(unique(cars2$car_company))
```

```
##  [1] amc        audi       bmw        buick      cadillac   capri
##  [7] chevrolet  chrysler   datsun     dodge      fiat       ford
## [13] hi         honda      mazda      mercedes   mercury    nissan
## [19] oldsmobile opel       peugeot    plymouth   pontiac    renault
## [25] saab       subaru     toyota     triumph    volkswagen volvo
## 30 Levels: amc audi bmw buick cadillac capri chevrolet chrysler ... volvo
```

```r
### looking at the data, i assume origin is a categorical data type, and its acceptable to assume horsep
#changing origin to character type, and horsepower to integer type
cars2$horsepower <- as.integer(cars2$horsepower)
```

```
## Warning: NAs introduced by coercion
```

```r
cars2$origin <- as.character(cars2$origin)

### i saw that after converting horsepower to integer, there are NA values in the horsepower variables c
# substitute all horsepower NA values with the mean horsepower value
is.na(cars2$horsepower)
```

```
##   [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [25] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
##  [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [97] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [121] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [145] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [169] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [181] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [193] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [205] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [217] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [229] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [241] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [253] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [265] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [277] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [301] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [313] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [325] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [337]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [349] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [361] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [373] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [385] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [397] FALSE FALSE
```

```r
cars2$horsepower[is.na(cars2$horsepower)] <- mean(cars2$horsepower, na.rm = TRUE)


### because of a error that arose due to unmatched data levels between training and testing data
### i gathered all car company categories that come up less than 3 times to a car company category call
# convert rare levels into "other" level

# Count the occurrences of each level
level_counts <- table(cars2$car_company)

# Identify rare levels
rare_levels <- names(level_counts[level_counts < 3]) #levels with a frequency below 3 will be grouped

# Replace rare levels with "Other"
cars2$car_company <- as.factor(ifelse(cars2$car_company %in% rare_levels,
                                      "other",
                                      as.character(cars2$car_company)))

# Check levels after modification
```

```r
levels(cars2$car_company)
```

```
##  [1] "amc"        "audi"       "buick"      "chevrolet"  "chrysler"
##  [6] "datsun"     "dodge"      "fiat"       "ford"       "honda"
## [11] "mazda"      "mercedes"   "mercury"    "oldsmobile" "opel"
## [16] "other"      "peugeot"    "plymouth"   "pontiac"    "renault"
## [21] "saab"       "subaru"     "toyota"     "volkswagen" "volvo"
```

```r
###splitting the data to training and testing splits before fitting it to regression models

library(rsample) # this library contains the training and testing functions
```

```
## Warning: package 'rsample' was built under R version 4.4.2
```

```r
set.seed(123) ### setting seed so that i can get same results if i do it again in the future
split <- initial_split(cars2, prop = 0.754) ###trying the best to split the data to 300 train and 98 te

train_data <- training(split)
test_data <- testing(split)

#creating a full multiple linear regression
cars_lm = lm(mpg ~ ., data = train_data)
summary(cars_lm)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6105 -2.3599  0.0271  1.8227 14.7348
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.492e+01  5.920e+00  -2.520  0.01232 *
## cylinder              -3.694e-01  4.013e-01  -0.921  0.35812
## displacement           2.543e-02  9.560e-03   2.660  0.00829 **
## horsepower            -3.219e-02  1.700e-02  -1.893  0.05940 .
## weight                -6.743e-03  8.146e-04  -8.278 6.05e-15 ***
## acceleration           3.311e-02  1.236e-01   0.268  0.78890
## model.year             7.389e-01  6.449e-02  11.458  < 2e-16 ***
## origin2               -1.027e+00  3.179e+00  -0.323  0.74685
## origin3                2.417e+00  3.971e+00   0.609  0.54331
## car_companyaudi        5.703e+00  3.525e+00   1.618  0.10691
## car_companybuick       1.095e+00  1.275e+00   0.859  0.39125
## car_companychevrolet   8.223e-01  9.806e-01   0.839  0.40245
## car_companychrysler    4.991e-01  2.590e+00   0.193  0.84733
## car_companydatsun      2.642e+00  4.112e+00   0.643  0.52105
## car_companydodge       1.794e+00  1.121e+00   1.601  0.11060
## car_companyfiat        5.426e+00  3.431e+00   1.581  0.11498
## car_companyford        4.093e-01  9.774e-01   0.419  0.67577
## car_companyhonda       2.515e+00  4.132e+00   0.609  0.54336
```

```
## car_companymazda       2.742e-01  4.167e+00   0.066  0.94759
## car_companymercedes    5.679e+00  3.889e+00   1.460  0.14543
## car_companymercury     1.761e-01  1.384e+00   0.127  0.89885
## car_companyoldsmobile  2.419e+00  1.464e+00   1.652  0.09979 .
## car_companyopel        3.549e+00  4.007e+00   0.886  0.37666
## car_companyother       3.257e+00  2.148e+00   1.516  0.13072
## car_companypeugeot     4.710e+00  3.566e+00   1.321  0.18764
## car_companyplymouth    2.373e+00  1.076e+00   2.206  0.02821 *
## car_companypontiac     3.600e+00  1.284e+00   2.804  0.00542 **
## car_companyrenault     6.091e+00  4.040e+00   1.508  0.13285
## car_companysaab        4.030e+00  3.800e+00   1.060  0.28993
## car_companysubaru      6.142e-01  4.365e+00   0.141  0.88820
## car_companytoyota      5.945e-01  4.100e+00   0.145  0.88482
## car_companyvolkswagen  5.845e+00  3.353e+00   1.743  0.08241 .
## car_companyvolvo       2.484e+00  3.609e+00   0.688  0.49198
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.376 on 267 degrees of freedom
## Multiple R-squared:  0.8396, Adjusted R-squared:  0.8204
## F-statistic: 43.67 on 32 and 267 DF,  p-value: < 2.2e-16
```

```
### equation: mpg ~ cylinder + displacement + horsepower + weight + acceleration + model.year + origin

#creating stepwise selection model (trying forward selection)
forward_lm <- step(cars_lm, direction = "forward")
```

```
## Start:  AIC=761.07
## mpg ~ cylinder + displacement + horsepower + weight + acceleration +
##     model.year + origin + car_company
```

```
summary(forward_lm)
```

```
##
## Call:
## lm(formula = mpg ~ cylinder + displacement + horsepower + weight +
##     acceleration + model.year + origin + car_company, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.6105 -2.3599  0.0271  1.8227 14.7348
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.492e+01  5.920e+00  -2.520  0.01232 *
## cylinder              -3.694e-01  4.013e-01  -0.921  0.35812
## displacement           2.543e-02  9.560e-03   2.660  0.00829 **
## horsepower            -3.219e-02  1.700e-02  -1.893  0.05940 .
## weight                -6.743e-03  8.146e-04  -8.278 6.05e-15 ***
## acceleration           3.311e-02  1.236e-01   0.268  0.78890
## model.year             7.389e-01  6.449e-02  11.458  < 2e-16 ***
## origin2               -1.027e+00  3.179e+00  -0.323  0.74685
## origin3                2.417e+00  3.971e+00   0.609  0.54331
```

```
## car_companyaudi        5.703e+00  3.525e+00   1.618  0.10691
## car_companybuick       1.095e+00  1.275e+00   0.859  0.39125
## car_companychevrolet   8.223e-01  9.806e-01   0.839  0.40245
## car_companychrysler    4.991e-01  2.590e+00   0.193  0.84733
## car_companydatsun      2.642e+00  4.112e+00   0.643  0.52105
## car_companydodge       1.794e+00  1.121e+00   1.601  0.11060
## car_companyfiat        5.426e+00  3.431e+00   1.581  0.11498
## car_companyford        4.093e-01  9.774e-01   0.419  0.67577
## car_companyhonda       2.515e+00  4.132e+00   0.609  0.54336
## car_companymazda       2.742e-01  4.167e+00   0.066  0.94759
## car_companymercedes    5.679e+00  3.889e+00   1.460  0.14543
## car_companymercury     1.761e-01  1.384e+00   0.127  0.89885
## car_companyoldsmobile  2.419e+00  1.464e+00   1.652  0.09979 .
## car_companyopel        3.549e+00  4.007e+00   0.886  0.37666
## car_companyother       3.257e+00  2.148e+00   1.516  0.13072
## car_companypeugeot     4.710e+00  3.566e+00   1.321  0.18764
## car_companyplymouth    2.373e+00  1.076e+00   2.206  0.02821 *
## car_companypontiac     3.600e+00  1.284e+00   2.804  0.00542 **
## car_companyrenault     6.091e+00  4.040e+00   1.508  0.13285
## car_companysaab        4.030e+00  3.800e+00   1.060  0.28993
## car_companysubaru      6.142e-01  4.365e+00   0.141  0.88820
## car_companytoyota      5.945e-01  4.100e+00   0.145  0.88482
## car_companyvolkswagen  5.845e+00  3.353e+00   1.743  0.08241 .
## car_companyvolvo       2.484e+00  3.609e+00   0.688  0.49198
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.376 on 267 degrees of freedom
## Multiple R-squared:  0.8396, Adjusted R-squared:  0.8204
## F-statistic: 43.67 on 32 and 267 DF,  p-value: < 2.2e-16
```

```
### equation: mpg ~ cylinder + displacement + horsepower + weight + acceleration + model.year + origin
###did not eliminate any variables

#trying backward selection
backward_lm <- step(cars_lm, direction = "backward")
```

```
## Start:  AIC=761.07
## mpg ~ cylinder + displacement + horsepower + weight + acceleration +
##     model.year + origin + car_company
##
##                Df Sum of Sq    RSS    AIC
## - car_company  24    336.63 3379.9 744.55
## - origin        2      7.76 3051.0 757.83
## - acceleration  1      0.82 3044.1 759.15
## - cylinder      1      9.66 3052.9 760.02
## <none>                      3043.3 761.07
## - horsepower    1     40.86 3084.1 763.07
## - displacement  1     80.65 3123.9 766.92
## - weight        1    781.07 3824.3 827.61
## - model.year    1   1496.35 4539.6 879.04
##
## Step:  AIC=744.55
## mpg ~ cylinder + displacement + horsepower + weight + acceleration +
```

```
##     model.year + origin
##
##                Df Sum of Sq    RSS    AIC
## - acceleration  1      1.42 3381.3 742.67
## - cylinder      1     14.06 3394.0 743.79
## <none>                      3379.9 744.55
## - horsepower    1     46.01 3425.9 746.60
## - displacement  1    127.08 3507.0 753.62
## - origin        2    283.10 3663.0 764.68
## - weight        1   1049.15 4429.0 823.65
## - model.year    1   1812.50 5192.4 871.35
##
## Step:  AIC=742.67
## mpg ~ cylinder + displacement + horsepower + weight + model.year +
##     origin
##
##                Df Sum of Sq    RSS    AIC
## - cylinder      1     14.38 3395.7 741.94
## <none>                      3381.3 742.67
## - horsepower    1     93.30 3474.6 748.84
## - displacement  1    125.66 3507.0 751.62
## - origin        2    282.52 3663.8 762.74
## - weight        1   1297.23 4678.5 838.09
## - model.year    1   1816.66 5198.0 869.67
##
## Step:  AIC=741.94
## mpg ~ displacement + horsepower + weight + model.year + origin
##
##                Df Sum of Sq    RSS    AIC
## <none>                      3395.7 741.94
## - horsepower    1     88.81 3484.5 747.69
## - displacement  1    133.89 3529.6 751.55
## - origin        2    274.48 3670.2 761.26
## - weight        1   1331.33 4727.0 839.18
## - model.year    1   1808.05 5203.7 868.01
```

```
summary(backward_lm)
```

```
##
## Call:
## lm(formula = mpg ~ displacement + horsepower + weight + model.year +
##     origin, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5429 -2.1777 -0.0065  1.9027 13.4328
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.525e+01  4.927e+00  -3.095  0.00216 **
## displacement  2.256e-02  6.638e-03   3.399  0.00077 ***
## horsepower   -3.375e-02  1.219e-02  -2.768  0.00599 **
## weight       -7.025e-03  6.555e-04 -10.718  < 2e-16 ***
## model.year    7.602e-01  6.086e-02  12.490  < 2e-16 ***
```

```
## origin2       2.844e+00  6.680e-01   4.258 2.79e-05 ***
## origin3       2.679e+00  6.405e-01   4.182 3.82e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.404 on 293 degrees of freedom
## Multiple R-squared:  0.821,  Adjusted R-squared:  0.8173
## F-statistic:   224 on 6 and 293 DF,  p-value: < 2.2e-16
```

```
### equation: mpg ~ displacement + horsepower + weight + model.year + origin
###adjusted R2, R2 , and residual standard error not improved but worsened
###eliminated car_company , acceleration, cylinder

#trying a stepwise both ways (backward and forward)
step_lm <- step(cars_lm, direction = "both")
```

```
## Start:  AIC=761.07
## mpg ~ cylinder + displacement + horsepower + weight + acceleration +
##     model.year + origin + car_company
##
##                Df Sum of Sq    RSS    AIC
## - car_company  24    336.63 3379.9 744.55
## - origin        2      7.76 3051.0 757.83
## - acceleration  1      0.82 3044.1 759.15
## - cylinder      1      9.66 3052.9 760.02
## <none>                      3043.3 761.07
## - horsepower    1     40.86 3084.1 763.07
## - displacement  1     80.65 3123.9 766.92
## - weight        1    781.07 3824.3 827.61
## - model.year    1   1496.35 4539.6 879.04
##
## Step:  AIC=744.55
## mpg ~ cylinder + displacement + horsepower + weight + acceleration +
##     model.year + origin
##
##                Df Sum of Sq    RSS    AIC
## - acceleration  1      1.42 3381.3 742.67
## - cylinder      1     14.06 3394.0 743.79
## <none>                      3379.9 744.55
## - horsepower    1     46.01 3425.9 746.60
## - displacement  1    127.08 3507.0 753.62
## + car_company  24    336.63 3043.3 761.07
## - origin        2    283.10 3663.0 764.68
## - weight        1   1049.15 4429.0 823.65
## - model.year    1   1812.50 5192.4 871.35
##
## Step:  AIC=742.67
## mpg ~ cylinder + displacement + horsepower + weight + model.year +
##     origin
##
##                Df Sum of Sq    RSS    AIC
## - cylinder      1     14.38 3395.7 741.94
## <none>                      3381.3 742.67
## + acceleration  1      1.42 3379.9 744.55
```

```
## - horsepower    1     93.30 3474.6 748.84
## - displacement  1    125.66 3507.0 751.62
## + car_company  24    337.23 3044.1 759.15
## - origin        2    282.52 3663.8 762.74
## - weight        1   1297.23 4678.5 838.09
## - model.year    1   1816.66 5198.0 869.67
##
## Step:  AIC=741.94
## mpg ~ displacement + horsepower + weight + model.year + origin
##
##                Df Sum of Sq    RSS    AIC
## <none>                      3395.7 741.94
## + cylinder      1     14.38 3381.3 742.67
## + acceleration  1      1.74 3394.0 743.79
## - horsepower    1     88.81 3484.5 747.69
## - displacement  1    133.89 3529.6 751.55
## + car_company  24    341.76 3053.9 758.12
## - origin        2    274.48 3670.2 761.26
## - weight        1   1331.33 4727.0 839.18
## - model.year    1   1808.05 5203.7 868.01
```

```
summary(step_lm)
```

```
##
## Call:
## lm(formula = mpg ~ displacement + horsepower + weight + model.year +
##     origin, data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5429 -2.1777 -0.0065  1.9027 13.4328
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.525e+01  4.927e+00  -3.095  0.00216 **
## displacement  2.256e-02  6.638e-03   3.399  0.00077 ***
## horsepower   -3.375e-02  1.219e-02  -2.768  0.00599 **
## weight       -7.025e-03  6.555e-04 -10.718  < 2e-16 ***
## model.year    7.602e-01  6.086e-02  12.490  < 2e-16 ***
## origin2       2.844e+00  6.680e-01   4.258 2.79e-05 ***
## origin3       2.679e+00  6.405e-01   4.182 3.82e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.404 on 293 degrees of freedom
## Multiple R-squared:  0.821,  Adjusted R-squared:  0.8173
## F-statistic:   224 on 6 and 293 DF,  p-value: < 2.2e-16
```

```
### equation: mpg ~ displacement + horsepower + weight + model.year + origin
###stepwise selection eliminated car_company, acceleration, cylinder just like backward selection
###but i will want to test it further with predictions and accuracy with test sample to see if its actu

### creating a SIMPLE linear regression with the variable that has the highest correlation to mpg
```

```r
#checking correlation bet mpg and all other variables.
#install.packages("GGally")
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.4.2
```

```
## Loading required package: ggplot2
```

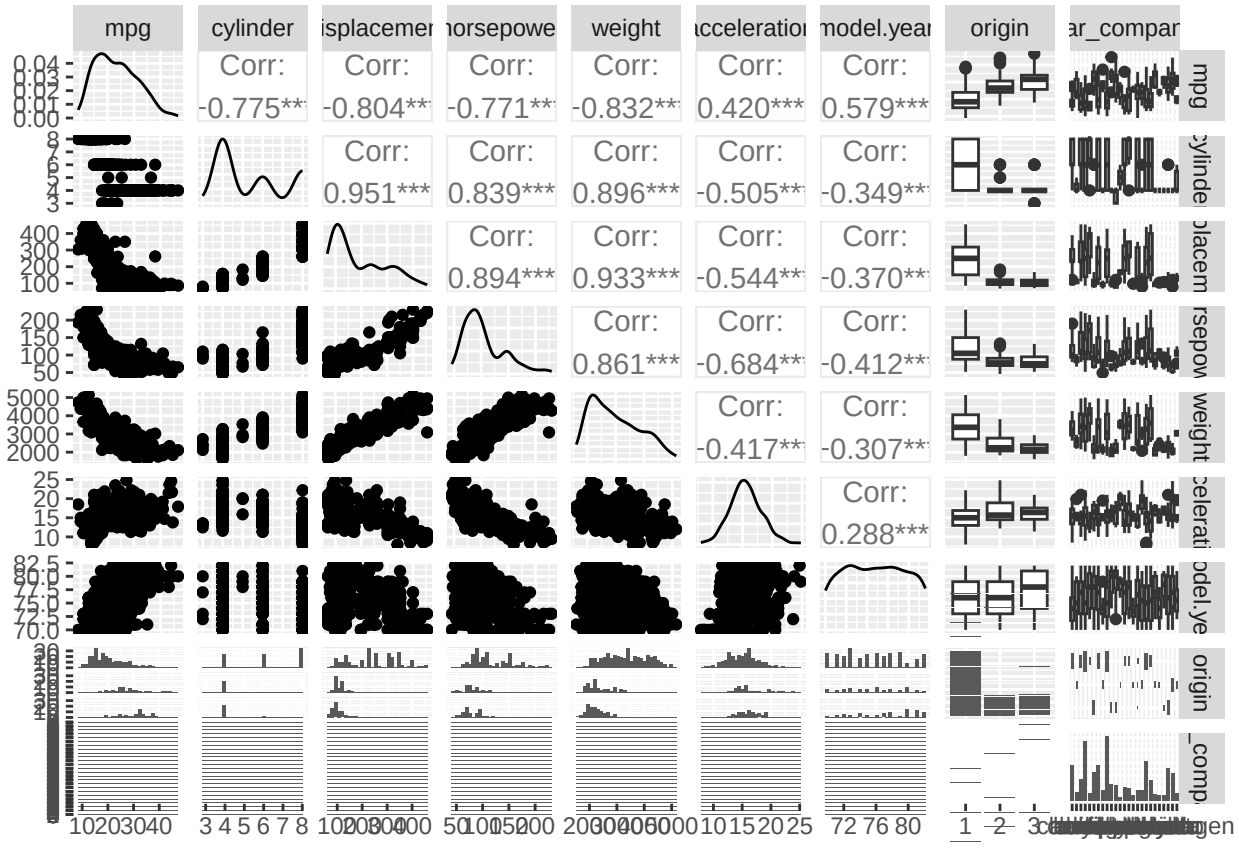
```
## Warning: package 'ggplot2' was built under R version 4.4.2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
ggpairs(cars2,cardinality_threshold = 25)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#the lsr library will help in finding correlation between mpg and the categorical variables like origin
library(lsr)
```

```
## Warning: package 'lsr' was built under R version 4.4.2
```

```
etaSquared(aov(cars2$mpg ~ cars2$car_company, data = cars2))
```

```
##                         eta.sq eta.sq.part
## cars2$car_company 0.3919308   0.3919308
```

```
### weight seems to have the highest correlation to mpg so ill create a linear regression bet mpg and w
weight_lm <- lm(mpg ~ weight, data = train_data)
summary(weight_lm)
```

```
##
## Call:
## lm(formula = mpg ~ weight, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12.2141  -2.7874  -0.5502   2.1071  16.2774
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 46.6842197  0.9154748   50.99   <2e-16 ***
## weight      -0.0077543  0.0002967  -26.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.398 on 298 degrees of freedom
## Multiple R-squared:  0.6962, Adjusted R-squared:  0.6952
## F-statistic: 682.9 on 1 and 298 DF,  p-value: < 2.2e-16
```

```
### the R2 significantly dropped compared to other models, yet we will use it for the sake of testing a

### one more multiple linear regression according to my queses and observations.
### i made some variables as a log() in the equation because i saw there are non linear relationships t
### i also eliminated origin since it gave me a higher R2 after its elimination
my_lm <- lm(mpg ~ cylinder + log(displacement) + log(horsepower) + log(weight) + log(acceleration) + mo
summary(my_lm)
```

```
##
## Call:
## lm(formula = mpg ~ cylinder + log(displacement) + log(horsepower) +
##     log(weight) + log(acceleration) + model.year + car_company,
##     data = train_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.4706 -1.8444  0.0417  1.4877 14.1409
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           137.07729   13.32270  10.289  < 2e-16 ***
## cylinder                0.58362    0.35688   1.635  0.10315
## log(displacement)      -0.28848    1.92648  -0.150  0.88108
## log(horsepower)        -8.98911    1.96748  -4.569 7.48e-06 ***
## log(weight)           -14.39765    2.89916  -4.966 1.22e-06 ***
## log(acceleration)      -5.48760    1.97275  -2.782  0.00579 **
## model.year              0.71314    0.05788  12.320  < 2e-16 ***
## car_companyaudi         3.34616    1.57867   2.120  0.03495 *
## car_companybuick        1.64133    1.15334   1.423  0.15586
## car_companychevrolet    0.21138    0.90041   0.235  0.81458
## car_companychrysler     1.21509    2.33982   0.519  0.60397
## car_companydatsun       3.45482    1.17487   2.941  0.00356 **
## car_companydodge        1.04943    1.02925   1.020  0.30883
## car_companyfiat         1.78950    1.49568   1.196  0.23258
## car_companyford        -0.09437    0.89655  -0.105  0.91625
## car_companyhonda        1.88522    1.32801   1.420  0.15689
## car_companymazda        0.84170    1.45827   0.577  0.56429
## car_companymercedes     3.41473    2.03962   1.674  0.09525 .
## car_companymercury     -0.81767    1.27083  -0.643  0.52050
## car_companyoldsmobile   2.40336    1.33758   1.797  0.07349 .
## car_companyopel         1.63407    2.34515   0.697  0.48654
## car_companyother        3.10328    1.50289   2.065  0.03989 *
## car_companypeugeot      3.17790    1.53439   2.071  0.03930 *
## car_companyplymouth     1.35405    0.98988   1.368  0.17249
## car_companypontiac      3.35881    1.14740   2.927  0.00371 **
```

```
## car_companyrenault      4.18780    2.38596   1.755  0.08037 .
## car_companysaab         3.18182    2.04526   1.556  0.12095
## car_companysubaru       1.27984    1.78601   0.717  0.47425
## car_companytoyota       1.33426    1.13313   1.178  0.24003
## car_companyvolkswagen   1.88118    1.23658   1.521  0.12937
## car_companyvolvo        0.62039    1.69961   0.365  0.71538
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.081 on 269 degrees of freedom
## Multiple R-squared:  0.8654, Adjusted R-squared:  0.8504
## F-statistic: 57.66 on 30 and 269 DF,  p-value: < 2.2e-16
```

```
### equation: mpg ~ cylinder + log(displacement) + log(horsepower) + log(weight) + log(acceleration) + r
### i got the best statistics through this regression, lets further assess with the prediction results

# predicting test sample on all five different models
full_pred <- predict(cars_lm, newdata = test_data)
step_pred <- predict(step_lm, newdata = test_data)
weight_pred <- predict(weight_lm, newdata = test_data)
my_pred <- predict(my_lm, newdata = test_data)
backward_pred <- predict(backward_lm, newdata = test_data)


# finding MAE of predictions of the three models
full_mae <- mean(abs(full_pred - test_data$mpg))
print(paste("Full MAE:", round(full_mae, 2)))
```

```
## [1] "Full MAE: 2.35"
```

```
step_mae <- mean(abs(step_pred - test_data$mpg))
print(paste("Step MAE:", round(step_mae, 2)))
```

```
## [1] "Step MAE: 2.41"
```

```
weight_mae <- mean(abs(weight_pred - test_data$mpg))
print(paste("Weight MAE:", round(weight_mae, 2)))
```

```
## [1] "Weight MAE: 3.25"
```

```
my_mae <- mean(abs(my_pred - test_data$mpg))
print(paste("My MAE:", round(my_mae, 2)))
```

```
## [1] "My MAE: 2.13"
```

```
backward_mae <- mean(abs(backward_pred - test_data$mpg))
print(paste("Backward MAE:", round(backward_mae, 2)))
```

```
## [1] "Backward MAE: 2.41"
```

```r
#finding RMSE
full_rmse <- sqrt(mean((full_pred - test_data$mpg)^2))
print(paste("Full RMSE:", round(full_rmse, 2)))
```

```
## [1] "Full RMSE: 2.97"
```

```r
step_rmse <- sqrt(mean((step_pred - test_data$mpg)^2))
print(paste("Step RMSE:", round(step_rmse, 2)))
```

```
## [1] "Step RMSE: 3.07"
```

```r
weight_rmse <- sqrt(mean((weight_pred - test_data$mpg)^2))
print(paste("Weight RMSE:", round(weight_rmse, 2)))
```

```
## [1] "Weight RMSE: 4.19"
```

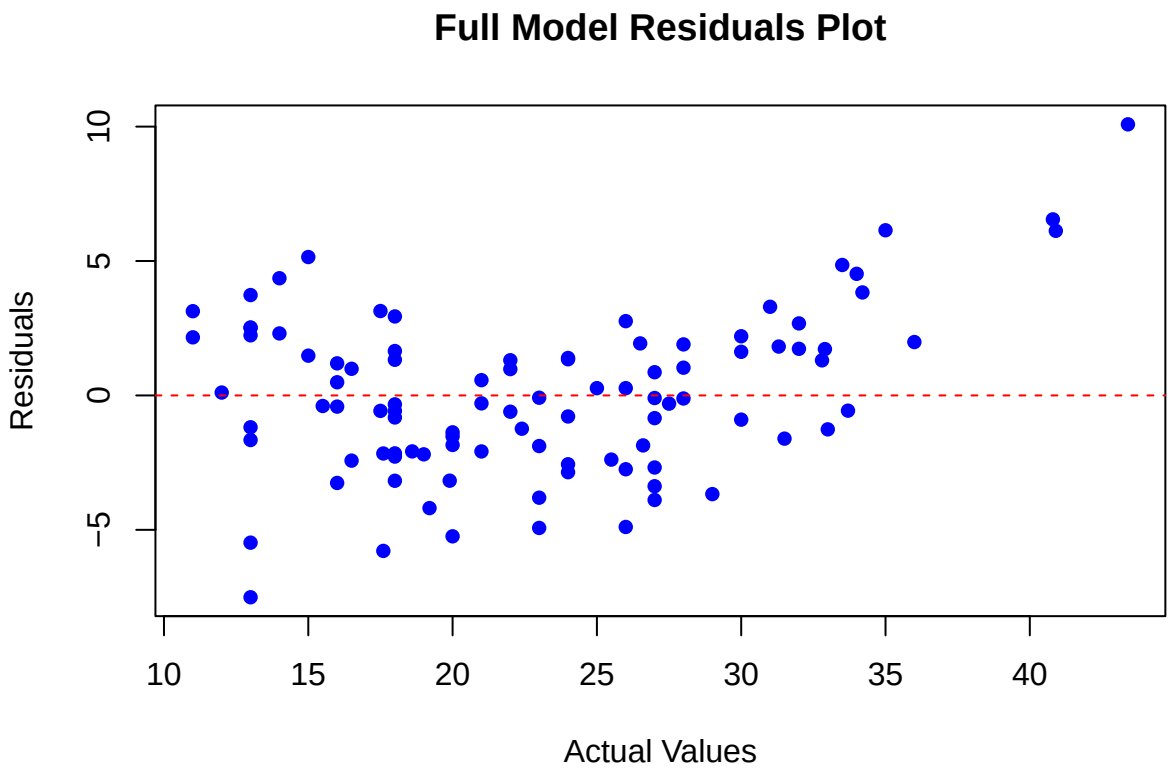
```r
my_rmse <- sqrt(mean((my_pred - test_data$mpg)^2))
print(paste("My RMSE:", round(my_rmse, 2)))
```

```
## [1] "My RMSE: 2.88"
```

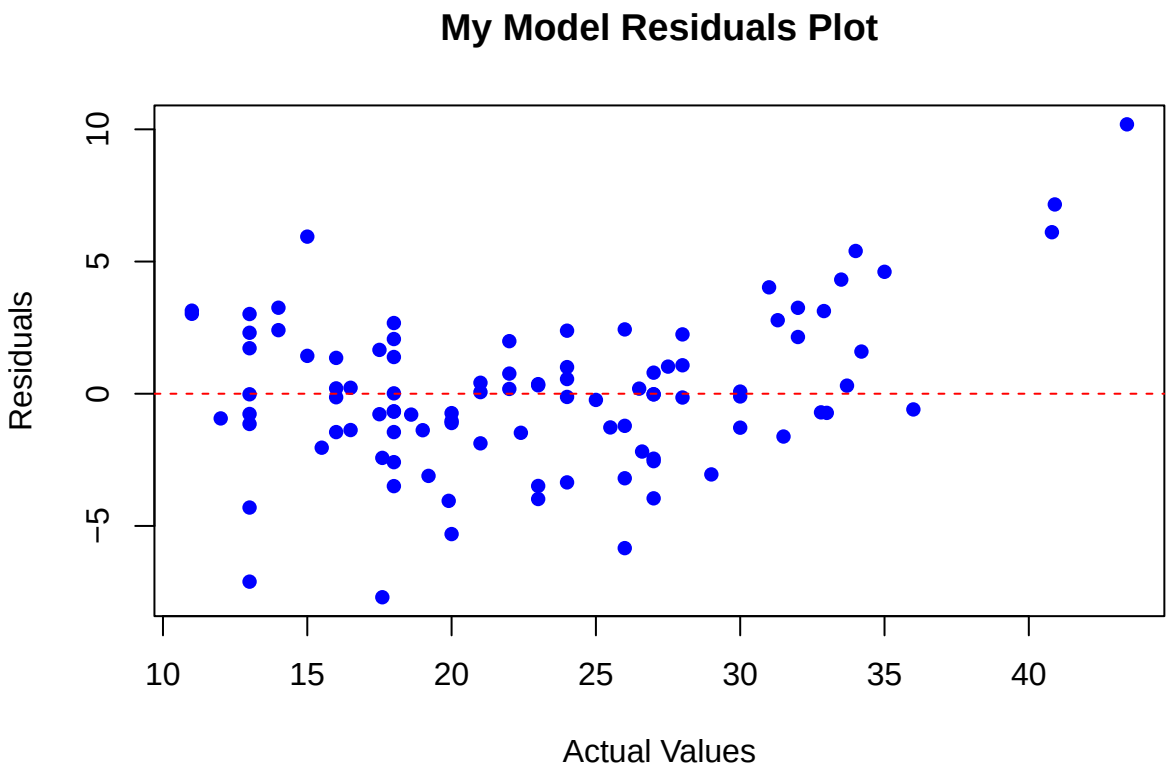
```r
backward_rmse <- sqrt(mean((backward_pred - test_data$mpg)^2))
print(paste("Backward RMSE:", round(backward_rmse, 2)))
```

```
## [1] "Backward RMSE: 3.07"
```

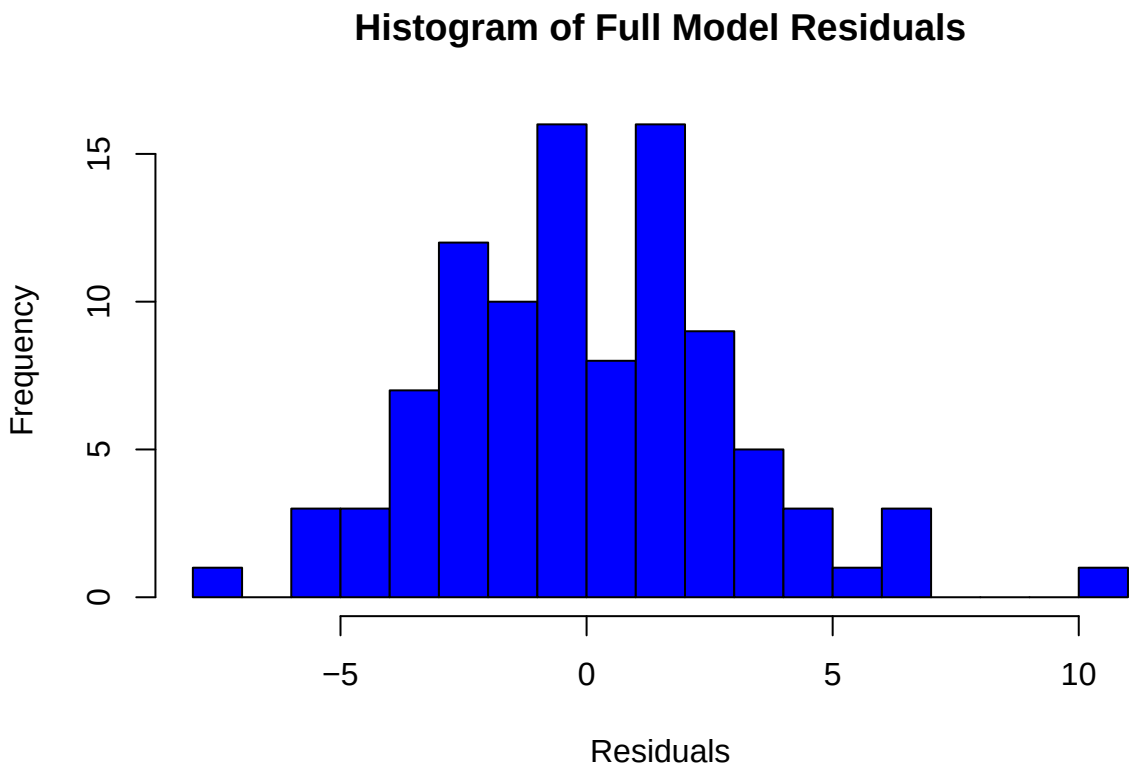
```r
# My model shows best results overall, with prediction and model fit
# in both MAE and RMSE this is the order best to worse: my, full, step/backward, weight

# lets bring in some residual plots and histogram to select 1 from the 2 best model which are so far th


# calculate residuals for my_pred (from the model i created) and full_pred (from the full model)
my_residuals <- test_data$mpg - my_pred
full_residuals <- test_data$mpg - full_pred


# Residuals plot
plot(test_data$mpg, full_residuals,
     xlab = "Actual Values",
     ylab = "Residuals",
     main = "Full Model Residuals Plot",
     col = "blue", pch = 16)
abline(h = 0, col = "red", lty = 2)  # Add a horizontal reference line at 0
```



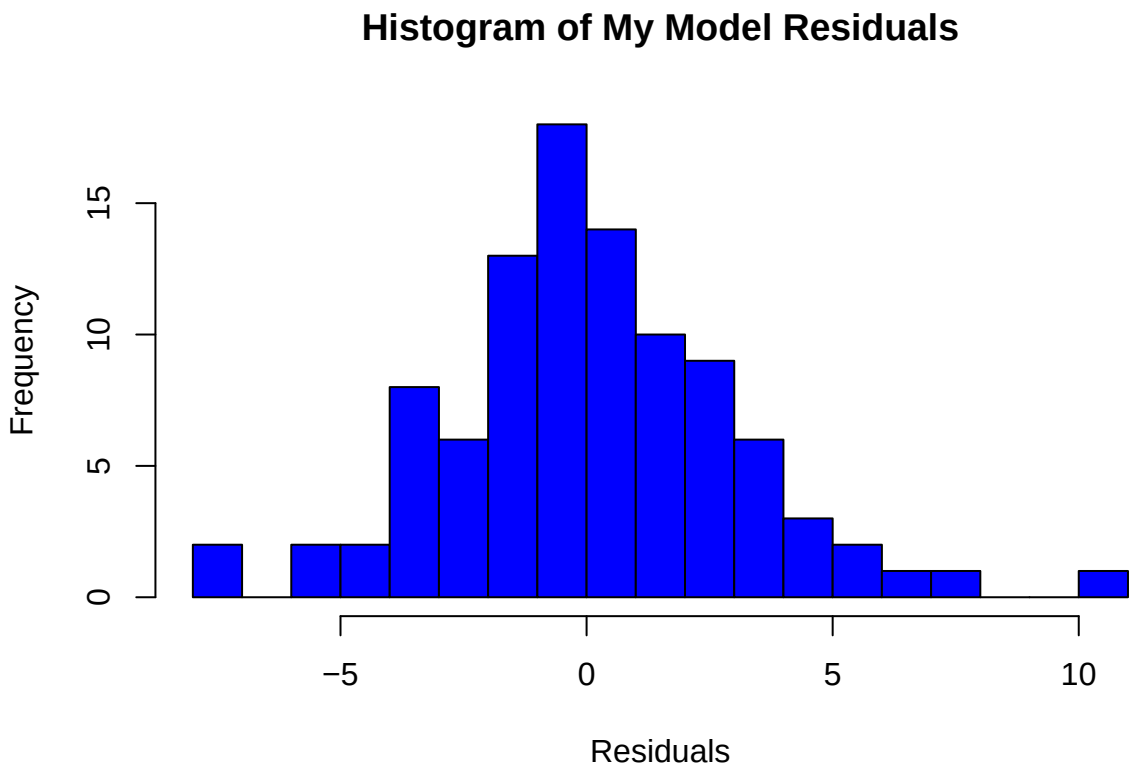

```r
# Residuals plot
plot(test_data$mpg, my_residuals,
     xlab = "Actual Values",
     ylab = "Residuals",
     main = "My Model Residuals Plot",
     col = "blue", pch = 16)
abline(h = 0, col = "red", lty = 2)  # Add a horizontal reference line at 0
```




```r
# Plot histogram for full_residuals (full model)
hist(full_residuals,
     main = "Histogram of Full Model Residuals",
     xlab = "Residuals",
     col = "blue",
     border = "black",
     breaks = 20)
```




```
# plot histogram for my_residuals (my model)
hist(my_residuals,
     main = "Histogram of My Model Residuals",
     xlab = "Residuals",
     col = "blue",
     border = "black",
     breaks = 20)
```




```
# looking at the histograms and residuals plots we see how the "my model" is performing better
# (histogram of the "my modle" residuals shows a better normal distribution and the residuals plot have
```