Data Analysis Final

Madhuri Raman

12/9/2021

Load in Required Packages and Data

```
library(tidyverse)
library(faraway)
library(MASS)
```

```
df <- read.csv("/Users/madhuri/Desktop/34300/auto.txt", sep = " ")
```

For this analysis, we are working with the autos data set which includes data about 205 different types of cars and records 22 different characteristics about each. One of these characteristics is `highway_mpg` which we aim to model using all of the other characteristics in this data set as covariates.

Data Cleaning

Check Data Types

```
df <- df %>% mutate_all(na_if, "?")
```

```
df$make <- as.factor(df$make) # 22
df$fuel_type <- as.factor(df$fuel_type) # 2
df$wheel_base <- as.numeric(df$wheel_base) #c
df$length <- as.numeric(df$length) #c
df$width <- as.numeric(df$width) #c
df$height <- as.numeric(df$height) #c
df$curb_weight <- as.numeric(df$curb_weight)
df$num_of_doors <- as.factor(df$num_of_doors) # 2
df$body_style <- as.factor(df$body_style) # 5
df$drive_wheels <- as.factor(df$drive_wheels) # 3
df$num_of_cylinders <- as.factor(df$num_of_cylinders) # 7
df$engine_type <- as.factor(df$engine_type) # 7
df$fuel_system <- as.factor(df$fuel_system) # 8
df$aspiration <- as.factor(df$aspiration) # 2
df$engine_size <- as.numeric(df$engine_size)
df$bore <- as.numeric(as.character(df$bore))
df$stroke <- as.numeric(as.character(df$stroke))
df$compression_rate <- as.numeric(df$compression_rate)
df$peak_rpm <- as.numeric(as.character(df$peak_rpm))
df$horsepower <- as.numeric(as.character(df$horsepower))
```

```r
df$normalized_losses <- as.numeric(as.character(df$normalized_losses))
df$highway_mpg <- as.numeric(df$highway_mpg)
```

We clean the data such that we can properly work with missing values, and we make sure that every covariate is encoded correctly as either a factor variable (categorical with several levels) or as a continuous variable (numeric).

Imputation of Missing Values

```r
# looking for the NAs and imputing them...

colSums(is.na(df))
```

```
##              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
```

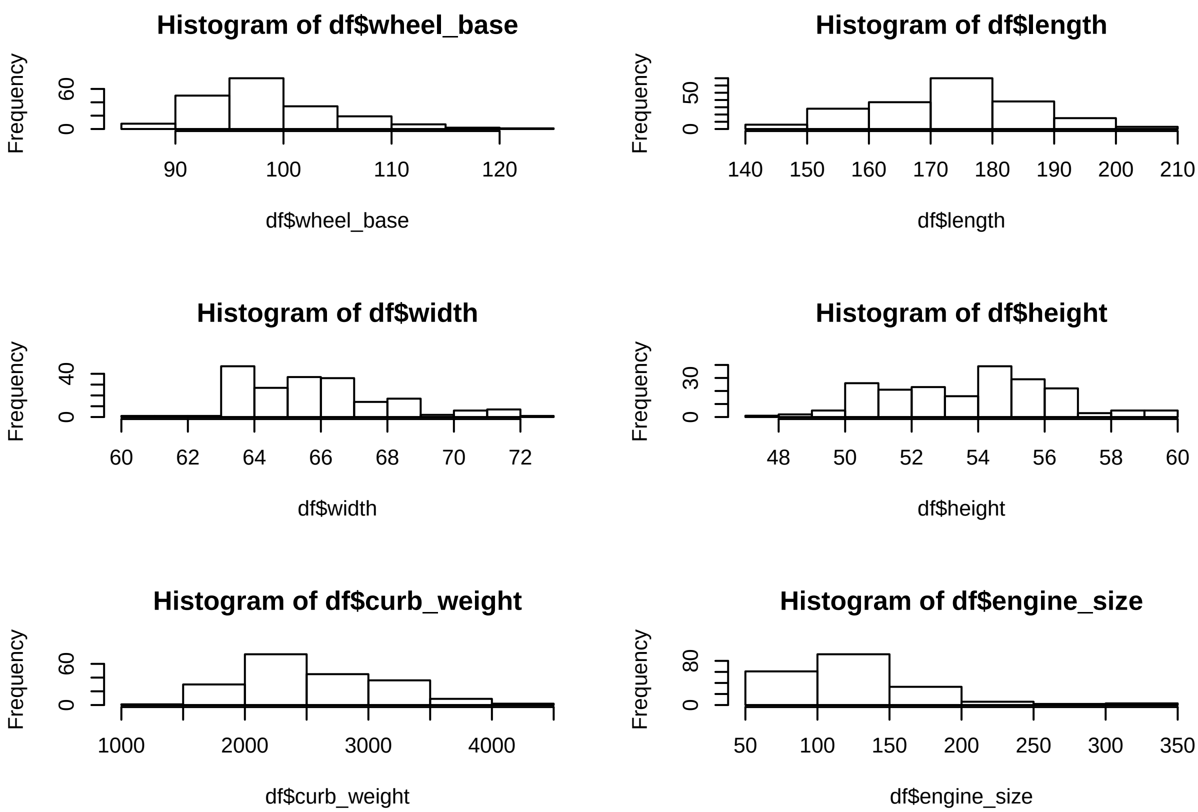
```r
# Mean Imputation for `normalized_losses`

nl_mean <- mean(df$normalized_losses, na.rm = TRUE)
df[is.na(df[,21]),21] <- rep(nl_mean, 41) # 21st col is `normalized_losses`

# Deletion for `num_of_doors`, `bore`, `stroke`, `peak_rpm`, `horsepower`

df <- na.omit(df)
```

First we need to identify the number of missing values in each column of the data frame. We see that `normalized_losses` has 41 missing values out of 205 observations which is a substantial amount of missing data. Also, there are 2 missing observations in `num_of_doors`, `peak_rpm`, and `horsepower` and 4 missing observations in `bore` and `stroke`. These rows with only 1 missing column (for the columns with 2 or 4 missing values total) can just be removed through deletion, as they do not impact the size of our data set very much. However, we would not want to remove all 41 rows with missing values for `normalized_losses` so we will use mean imputation to fill in the missing values for `normalized_losses`. Although mean imputation is good here because there are a lot of missing values within one column, we must acknowledge that it will create some bias that may not be compensated by reduction in variance and keep this mind when we begin the modeling stage.

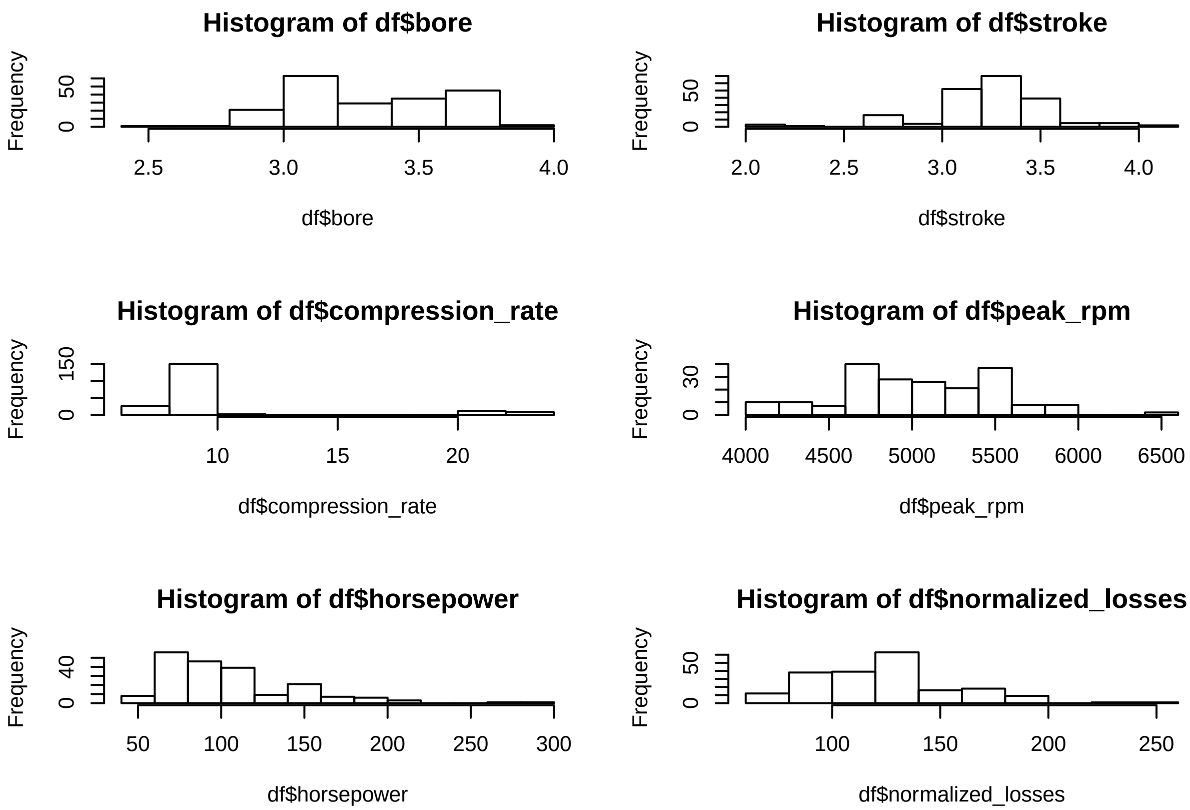
EDA

First we will look at histograms of our covariates and response to get a sense for the distribution of each variables. In general, we are curious about which variables look normally distributed in the data and which may be more skewed.

```r
par(mfrow=c(3,2))
hist(df$wheel_base)
hist(df$length)
hist(df$width)
hist(df$height)
hist(df$curb_weight)
hist(df$engine_size)
```

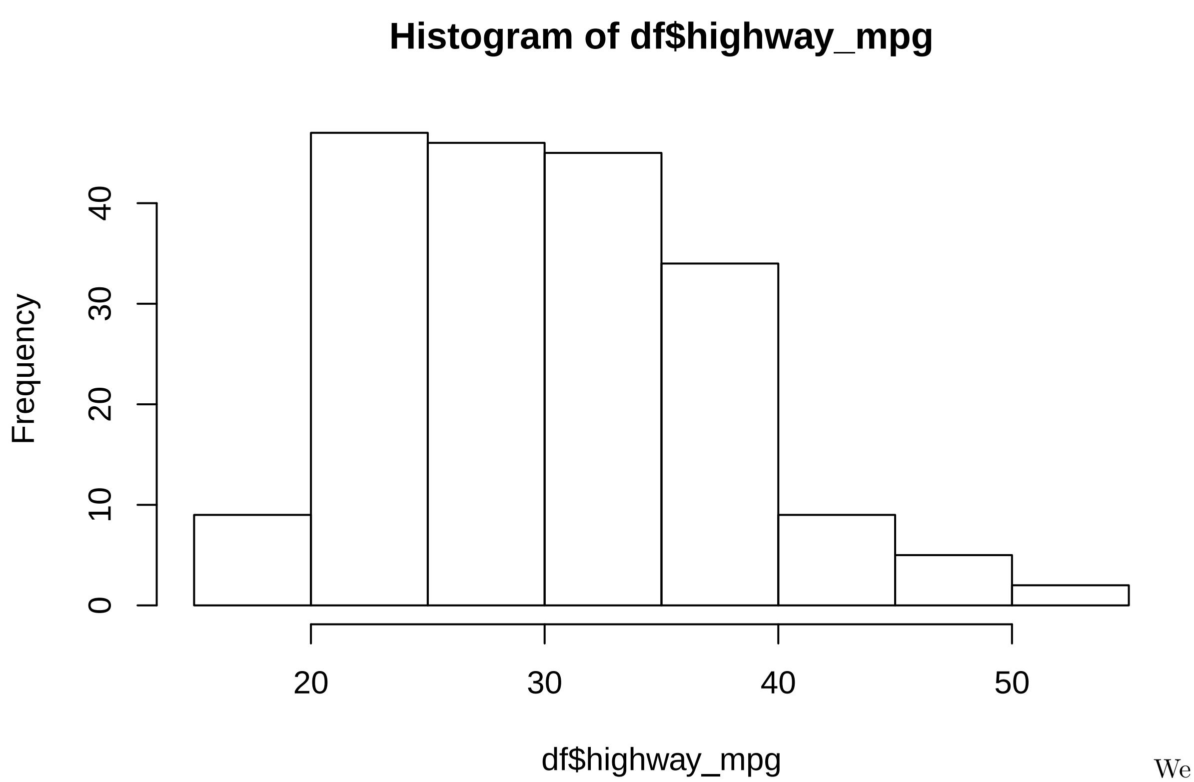



```r
par(mfrow=c(3,2))

hist(df$bore)
hist(df$stroke)
hist(df$compression_rate)
hist(df$peak_rpm)
hist(df$horsepower)
hist(df$normalized_losses)
```



Notably, the distributions of `compression_rate`, `engine_size`, and `horsepower` look fairly skewed right, while the distributions of `length` and `normalized_losses`, for example, look quite normal/bell-curve shaped.

```
hist(df$highway_mpg)
```



We also

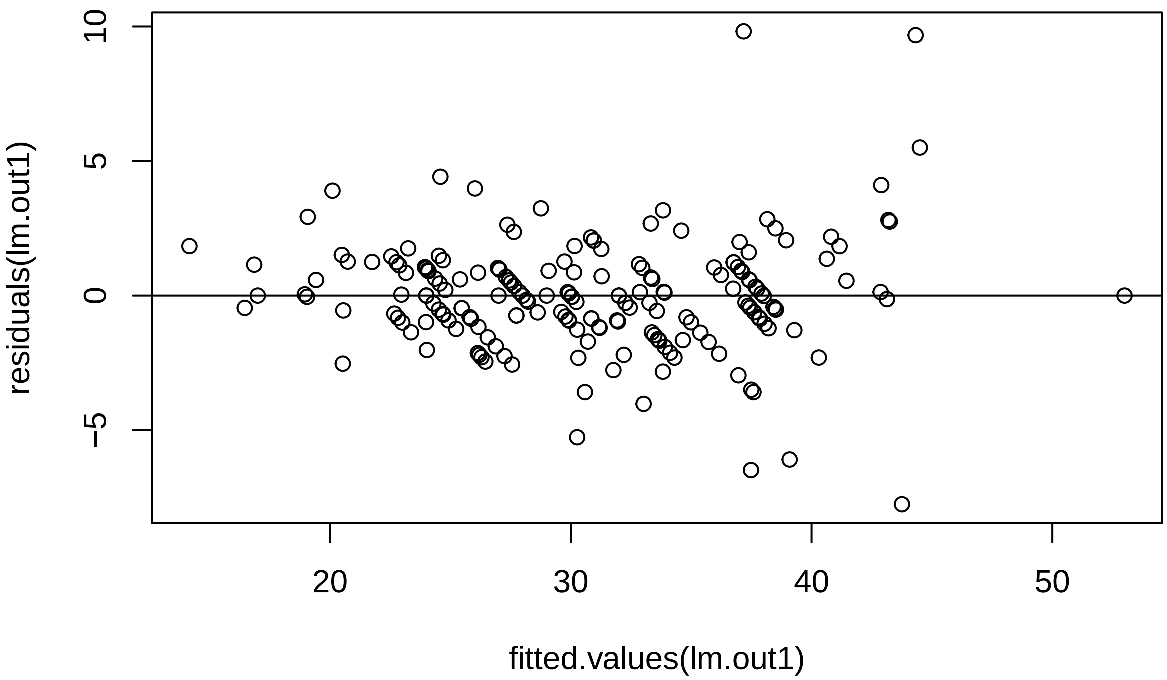
should note that the distribution of our response `highway_mpg` looks very slightly skewed to the right but generally is unimodal centered around 30. We should think about transforming the response if we end up seeing issues with the residuals vs. fitted values plot in our model.

Initial Full Regression Model

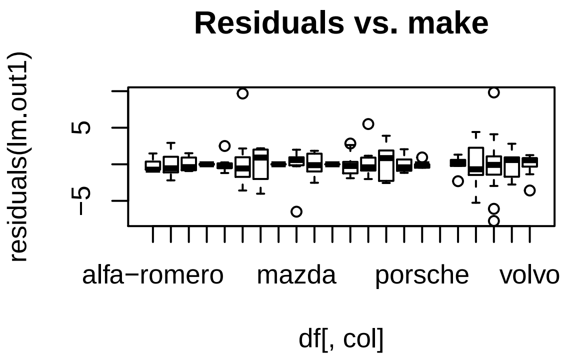
```
lm.out1 <- lm(highway_mpg ~., data = df)
# summary(lm.out1)
```

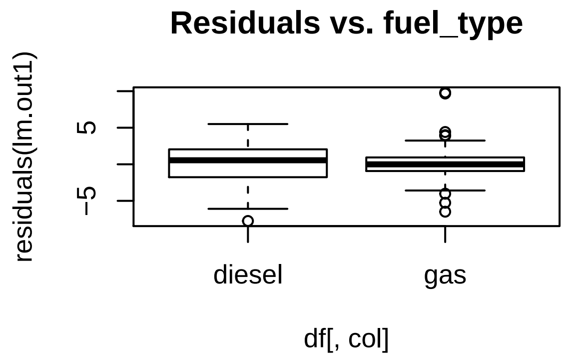
Check Residual Plots

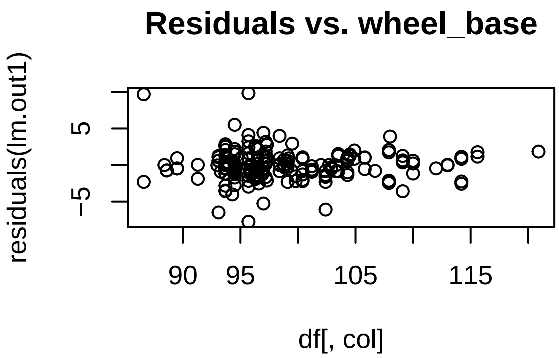
```
plot(residuals(lm.out1) ~ fitted.values(lm.out1))
abline(h=0)
```

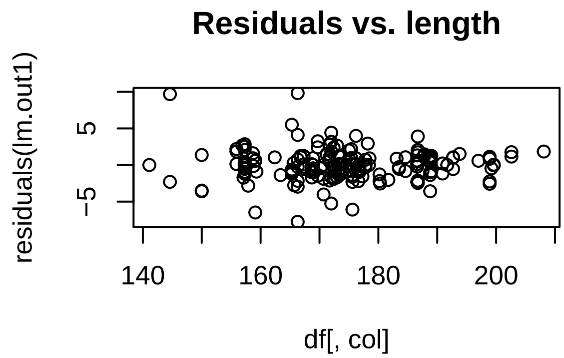


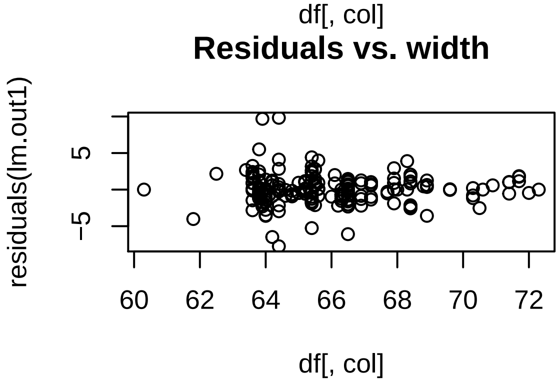

```
par(mfrow=c(2,2))
for (col in 1:21){
  plot(residuals(lm.out1) ~ df[,col], main=paste("Residuals vs.", colnames(df)[col]))
}
```

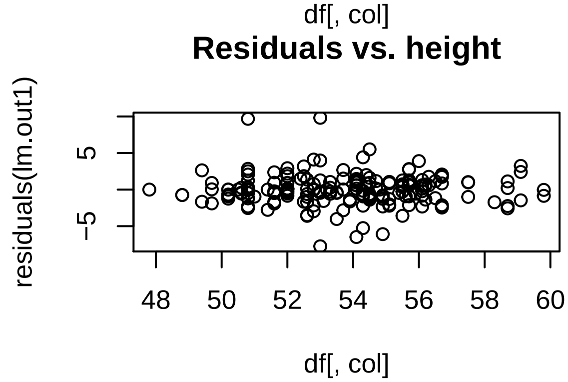

Residuals vs. make
residuals(lm.out1)
5
−5
alfa−romero
mazda
porsche
volvo
df[, col]

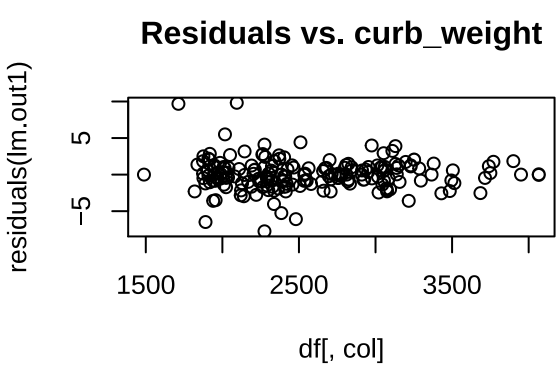

Residuals vs. fuel_type
residuals(lm.out1)
5
−5
diesel
gas
df[, col]

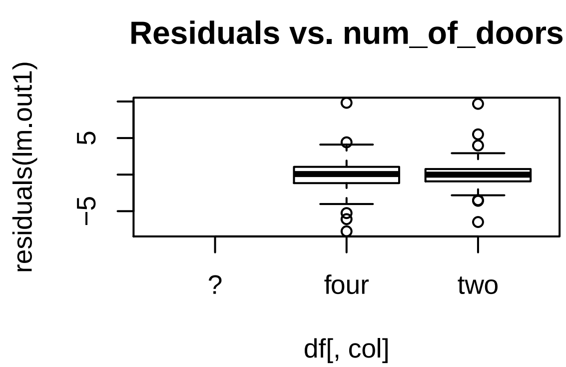

Residuals vs. wheel_base
residuals(lm.out1)
5
−5
90
95
105
115
df[, col]

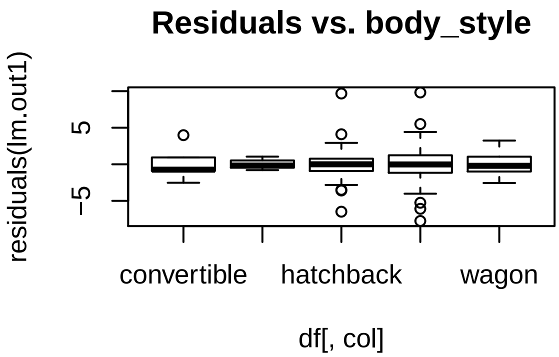

Residuals vs. length
residuals(lm.out1)
5
−5
140
160
180
200
df[, col]

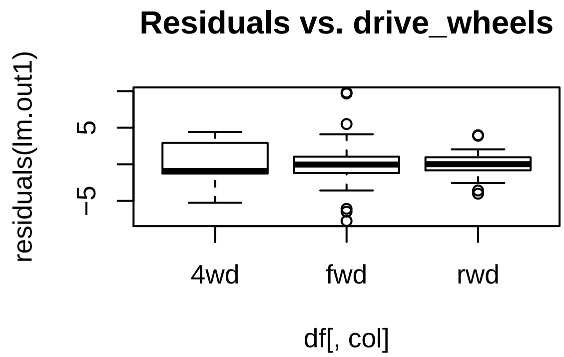

Residuals vs. width
residuals(lm.out1)
5
−5
60
62
64
66
68
70
72
df[, col]

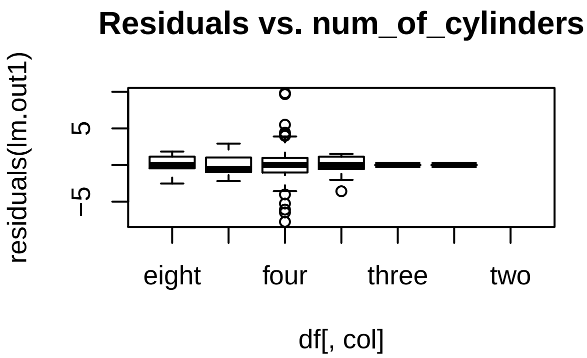

Residuals vs. height
residuals(lm.out1)
5
−5
48
50
52
54
56
58
60
df[, col]

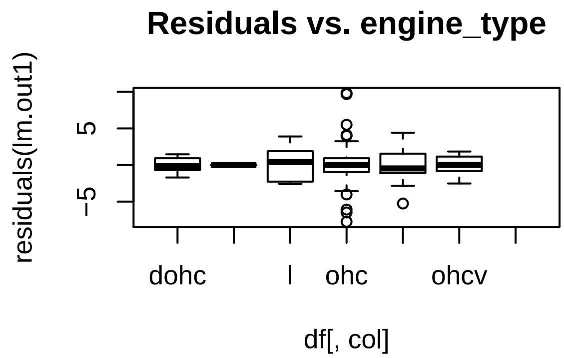

Residuals vs. curb_weight
residuals(lm.out1)
5
−5
1500
2500
3500
df[, col]

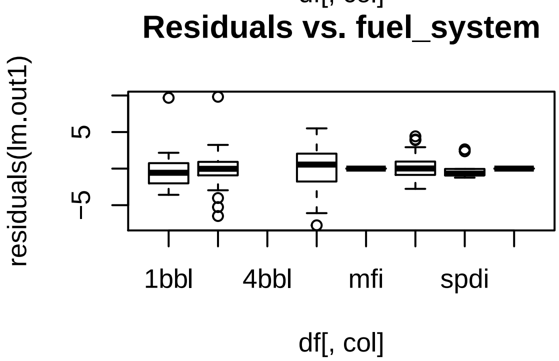

Residuals vs. num_of_doors
residuals(lm.out1)
5
−5
?
four
two
df[, col]

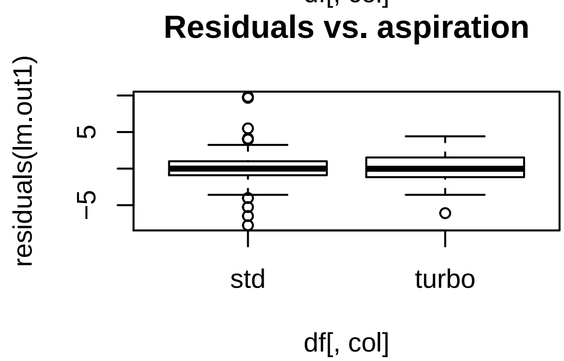

Residuals vs. body_style
residuals(lm.out1)
5
−5
convertible
hatchback
wagon
df[, col]

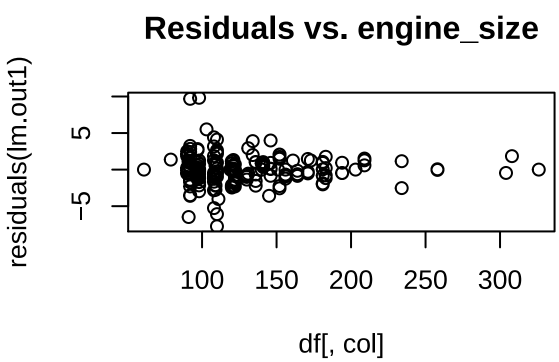

Residuals vs. drive_wheels
residuals(lm.out1)
5
−5
4wd
fwd
rwd
df[, col]

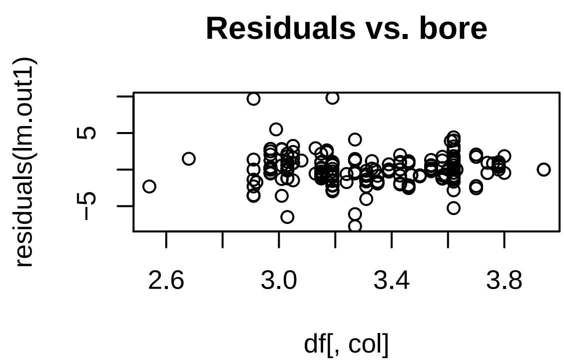

Residuals vs. num_of_cylinders
residuals(lm.out1)
5
−5
eight
four
three
two
df[, col]

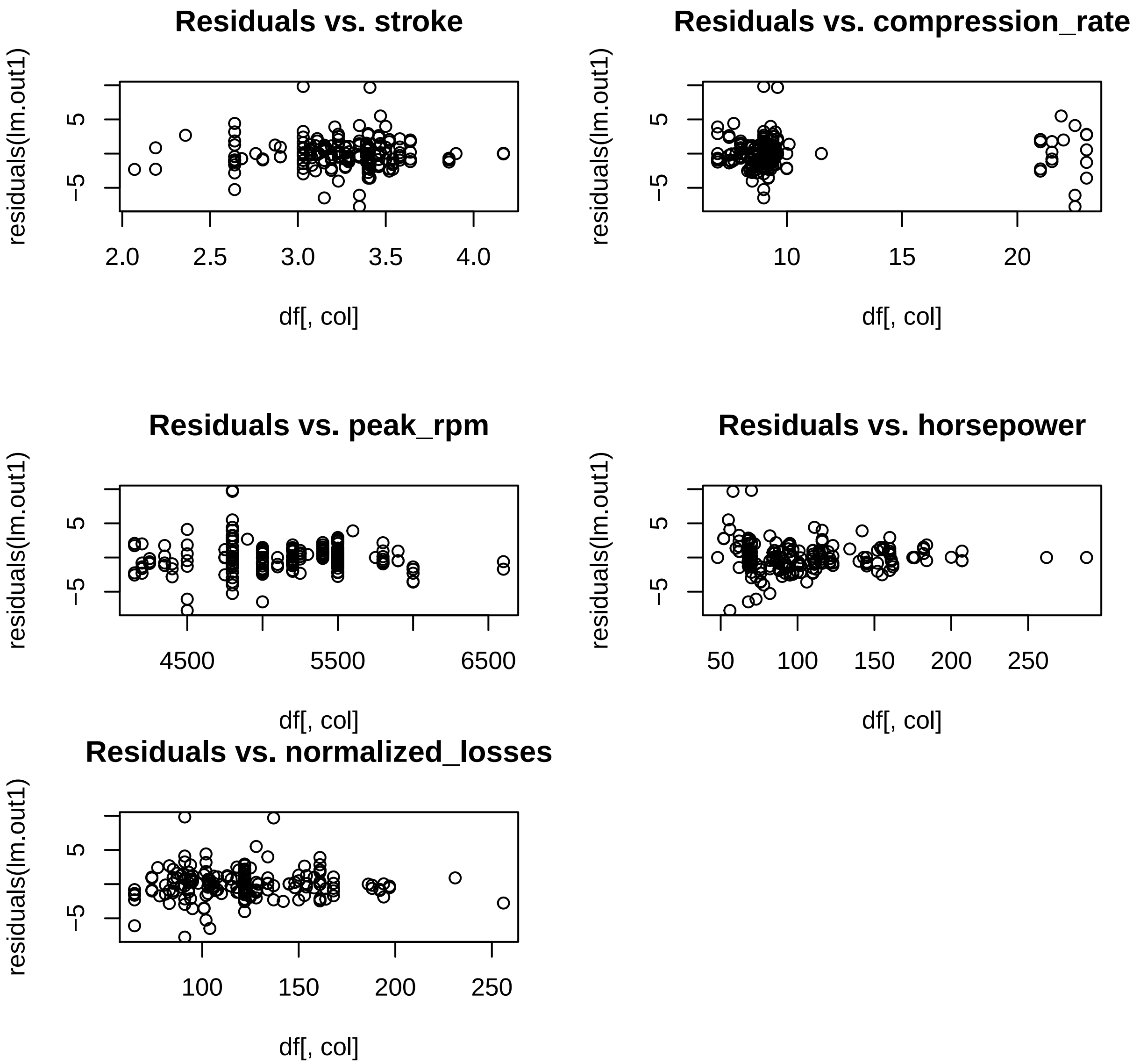

Residuals vs. engine_type
residuals(lm.out1)
5
−5
dohc
l
ohc
ohcv
df[, col]


Residuals vs. fuel_system
residuals(lm.out1)
5
−5
1bbl
4bbl
mfi
spdi
df[, col]


Residuals vs. aspiration
residuals(lm.out1)
5
−5
std
turbo
df[, col]


Residuals vs. engine_size
residuals(lm.out1)
5
−5
100
150
200
250
300
df[, col]


Residuals vs. bore
residuals(lm.out1)
5
−5
2.6
3.0
3.4
3.8
df[, col]



We examine the residual plots vs. fitted values and vs. each covariate and want to identify potentially high leverage and outlier points, nonconstant variance issues, or nonlinearity issues. The residuals vs. fitted values plot looks pretty good; there are no obvious issues with it other than a couple potential outliers with high residuals of around 10 and a potentially high leverage point with high fitted value above 50. Specifically, there are potentially high leverage points at the extreme right tails of the residual plots vs. `horsepower`, and vs. `engine_size`, for example. There also looks like there may be nonconstant variance between levels for some of the categorical variables, specifically for `body_style`, `fuel_system`, and `aspiration`, so we should keep an eye on if these variance improve in our future models. Another notable characteristic is the very clear clustering behavior of the residuals vs. `compression_rate`. However this makes sense in the context of the problem since, as we see below, the high values of `compression_rate` around 20 are clearly associated with diesel `fuel_type` and idi `fuel_system` while the low values of `compression_rate` around 10 are directly associated with gas `fuel_type` other types of `fuel_system`. We must consider these direct associations when building our model.

```
dftmp <- df[,c(18,13,2)]
head(dftmp[order(dftmp$compression_rate, decreasing = TRUE),], n = 10)
```

```
##     compression_rate fuel_system fuel_type
## 183             23.0         idi    diesel
```

```
## 185              23.0         idi    diesel
## 188              23.0         idi    diesel
## 193              23.0         idi    diesel
## 204              23.0         idi    diesel
## 159              22.5         idi    diesel
## 160              22.5         idi    diesel
## 175              22.5         idi    diesel
## 67               22.0         idi    diesel
## 91               21.9         idi    diesel
```

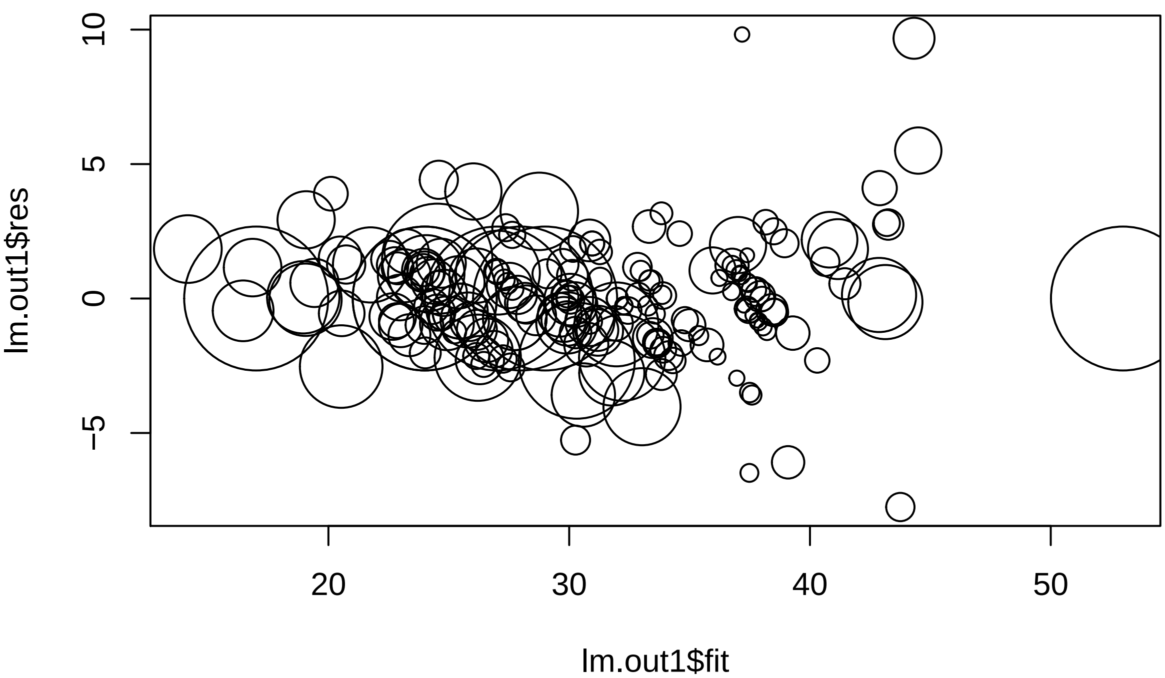
```
tail(dftmp[order(dftmp$compression_rate, decreasing = TRUE),], n = 10)
```

```
##     compression_rate fuel_system fuel_type
## 89               7.5        spdi       gas
## 199              7.5        mpfi       gas
## 200              7.5        mpfi       gas
## 10               7.0        mpfi       gas
## 30               7.0         mfi       gas
## 83               7.0        spdi       gas
## 84               7.0        spdi       gas
## 85               7.0        spdi       gas
## 118              7.0        mpfi       gas
## 125              7.0        spdi       gas
```

Check for Leverage Points

Let's check to see if there are any unusually high leverage points.

```
lev = hatvalues(lm.out1)
plot(lm.out1$fit,lm.out1$res,cex=10*lev)
```



We see that the suspicious point we identified earlier with fitted value greater than 50 has high leverage but not any higher leverage than many of the points around fitted value of 30. Also, the two points we identified with high residuals around 10 actually have low leverage on this model, which is good news. This suggests that we do not need to remove any specific high leverage points or outliers.

Model Selection

Check for Collinearity

Model 1

Based on the clustering behavior of the residuals vs. `compression_rate` in the full model and our investigation into the cause of these clusters, we know that there are some covariates in the model that are extremely collinear. Linear regression does not perform well when there is a high amount of collinearity between covariates, so we will examine the variance inflation factor (VIF) scores to understand the degree to which collinearity is affected our predictors.

```r
sort(vif(model.matrix(lm.out1)[,-1]), decreasing = TRUE)
```

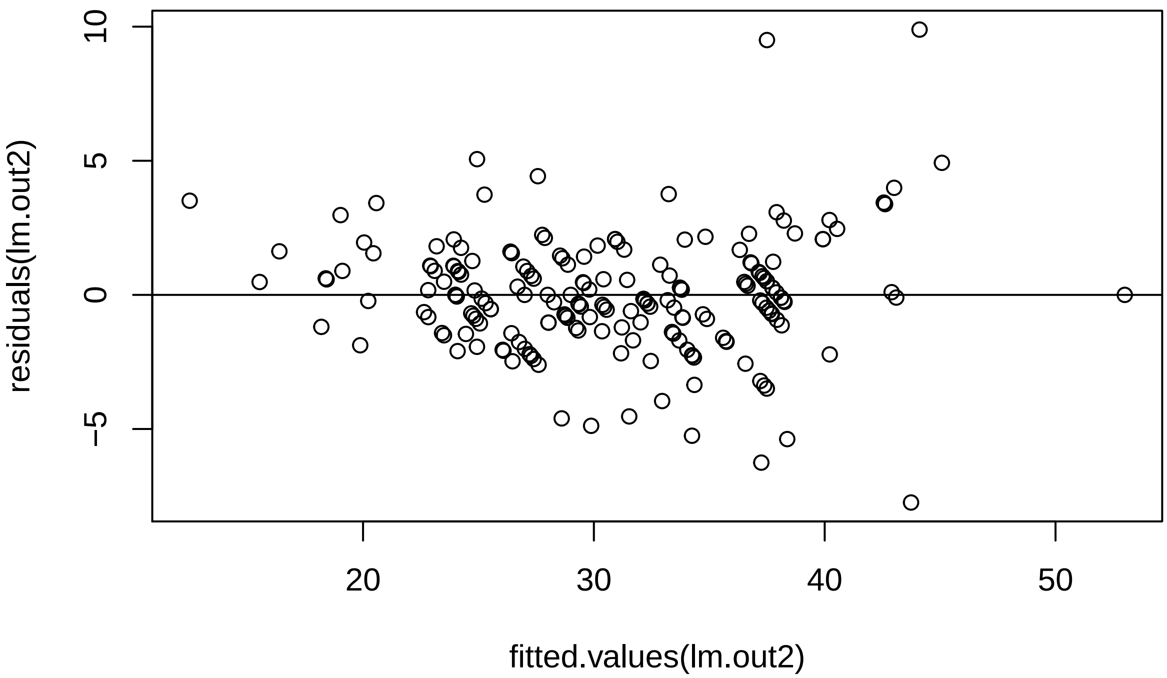
```
##             makepeugot           fuel_typegas  num_of_cylindersthree
##                    Inf                    Inf                    Inf
##            engine_typel         fuel_systemidi       compression_rate
##                    Inf                    Inf             225.359744
##    num_of_cylindersfour            engine_size             horsepower
##             133.917058              67.364649              59.722839
##     num_of_cylinderssix            curb_weight             makesubaru
##              48.649960              48.618994              43.612581
##         fuel_systemmpfi         engine_typeohcf        fuel_system2bbl
##              37.782412              32.741197              30.522536
##    num_of_cylindersfive                 length         body_stylesedan
##              27.747163              24.108333              22.532450
##             maketoyota         drive_wheelsrwd              makehonda
##              21.752756              20.019664              18.855539
##          engine_typeohc             wheel_base    body_stylehatchback
##              18.702066              18.570832              17.128792
##                   bore             makenissan      makemercedes-benz
##              16.489778              16.378915              15.991706
##              makevolvo         makemitsubishi            makeporsche
##              15.519644              14.706919              14.678810
##                makebmw                  width         makevolkswagen
##              14.601478              14.532330              12.767590
##               makeaudi        drive_wheelsfwd              makemazda
##              12.392797              12.180655              11.489888
##         body_stylewagon         fuel_systemspdi              makedodge
##              11.218454               9.151880               9.057379
##               makesaab num_of_cylinderstwelve                 height
##               8.744851               8.321774               7.760152
##           makeplymouth             makejaguar       engine_typedohcv
##               7.676497               7.375906               7.179593
##                 stroke        aspirationturbo        engine_typeohcv
##               6.197340               6.012789               5.997036
##               peak_rpm              makeisuzu          makechevrolet
##               5.525319               4.994687               4.326791
##      normalized_losses     body_stylehardtop          num_of_doorstwo
##               3.717373               3.542027               3.426751
```

```
##           makemercury         fuel_systemspfi          fuel_systemmfi
##              2.835695                2.286554                2.216256
```

Based on the VIF scores from the full model, we see that `fuel_type`, `compression_rate`, and `num_of_cylinders` for several levels have extremely high or even infinite scores. This means that their variances is the most inflated due to collinearity compared to what their variances would be if they were orthogonal covariates. We know from the context of the problem that `fuel_type` and `compression_rate` are very collinear, and we also see that `num_of_cylinders` for most of its levels has very high VIF scores. We will rerun our regression without these three variables and check to see if the VIF scores have improved overall.

Model 2

```r
lm.out2 <- lm(highway_mpg ~ . - compression_rate - num_of_cylinders - fuel_type, data = df)

plot(residuals(lm.out2) ~ fitted.values(lm.out2))
abline(h=0)
```




```r
sort(vif(model.matrix(lm.out2)[,-1]), decreasing = TRUE)
```

```
##          makesubaru          horsepower         engine_size      fuel_systemmpfi
##           41.097110           38.127475           37.882127            36.950558
##         curb_weight          makepeugot      engine_typeohcf      fuel_system2bbl
##           34.704519           31.288931           30.527736            30.305685
##              length        engine_typel      body_stylesedan      drive_wheelsrwd
##           22.062429           21.645806           21.180039            17.988859
##          wheel_base           makehonda           maketoyota body_stylehatchback
##           17.713891           17.232479           17.214525            15.928064
##      fuel_systemidi               width          makenissan       makemitsubishi
##           15.151269           13.728316           13.518692            13.338552
##         makeporsche    drive_wheelsfwd           makevolvo       body_stylewagon
##           11.446064           11.266198           10.780574            10.708566
```

```
##      makevolkswagen   makemercedes-benz      engine_typeohc             makemazda
##           10.498631            9.945083            9.484441              9.457790
##           makeaudi      fuel_systemspdi             makebmw             makedodge
##            8.835716            8.655104            8.647886              8.147285
##              height        makeplymouth            makesaab                  bore
##            7.192450            6.958220            6.815186              5.894755
##     aspirationturbo          makejaguar            peak_rpm       engine_typeohcv
##            5.453563            5.207181            5.087709              5.054355
##           makeisuzu              stroke       makechevrolet  normalized_losses
##            4.469216            4.435887            3.860857              3.683676
##     num_of_doorstwo     engine_typedohcv   body_stylehardtop       fuel_systemspfi
##            3.366430            3.324692            3.243887              2.237088
##         makemercury       fuel_systemmfi
##            2.164661            2.153105
```

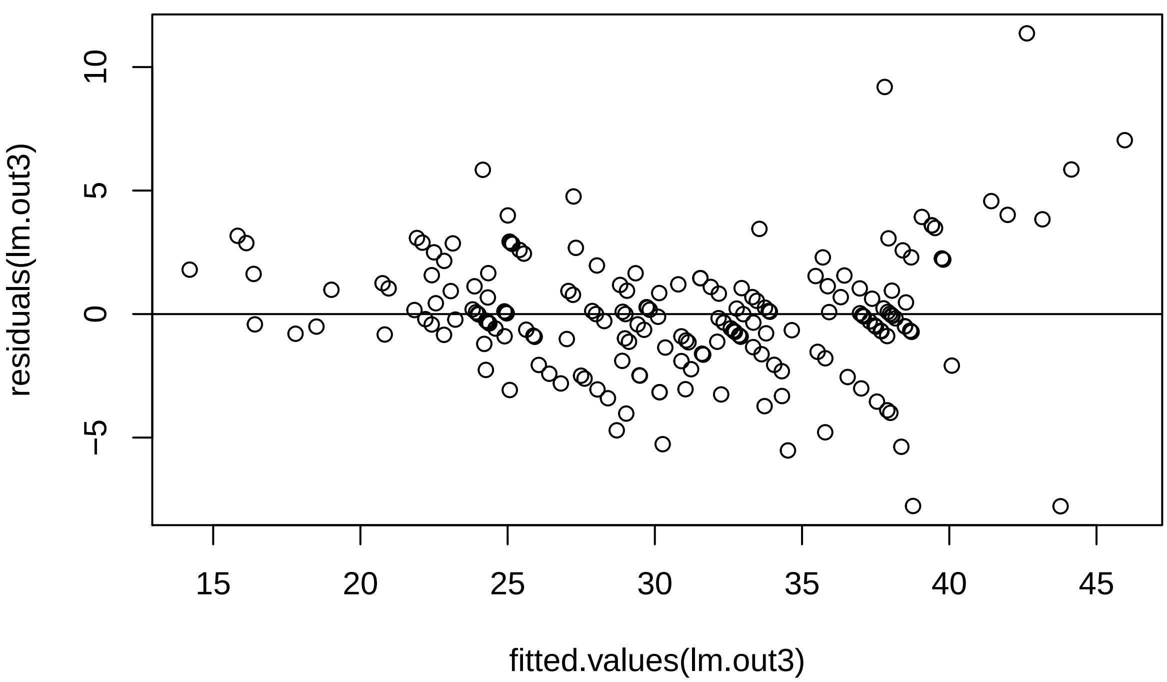
In model 2 we see very little affect of removing the three covariates on the residuals vs. fitted values, which is a good sign because it means that removing these three variables did not create any new problems to model correctness. Reevaluating the VIF scores, we see that the variance of `make` is also very inflated due to collinearity. This covariate also has 22 different levels, and it does not make sense that differences among these 22 levels would contribute to `highway_mpg` so we will rerun the model one more time without `make` before we proceed with backwards elimination.

Model 3

```r
colnames(df[,c(1,2,11,18)]) # these are the indices of the covariates we removed
```

```
## [1] "make"              "fuel_type"         "num_of_cylinders" "compression_rate"
```

```r
lm.out3 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18)])

plot(residuals(lm.out3) ~ fitted.values(lm.out3))
abline(h=0)
```



Above we ran model 3, the linear regression model with all covariates except for `compression_rate`, `num_of_cylinders`, `fuel_type`, and `make`. We confirm again that the residuals vs. fitted values plot still looks pretty good.

Backward Elimination to Remove Non-significant Covariates

Starting with model 3, we will perform backwards elimination by p-value significance to remove covariates that are not significant in predicting the response `highway_mpg`. We will keep repeating this process until all of our p-values are significant at an alpha level threshold of 0.10.

```r
# continuous covariates are indices: c(2:5, 26:31)
# categorical covariates are: num_of_doors, body_style, drive_wheels, engine_type,
# fuel_system, aspiration
# their indices are c(8, 9, 10, 12, 13, 14)

# Model 3 continuous covariates' p-values:

round(summary(lm.out3)$coef[c(2:6, 26:31),4],3)
```

```
##        wheel_base            length             width            height
##             0.383             0.016             0.812             0.253
##       curb_weight       engine_size              bore            stroke
##             0.001             0.778             0.856             0.471
##          peak_rpm        horsepower normalized_losses
##             0.069             0.451             0.137
```

```r
# p num_of_doors
anova(lm.out3, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,8)]))$Pr[2]
```

```
## [1] 0.7430172
```

```r
# p body_style
anova(lm.out3, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,9)]))$Pr[2]
```

```
## [1] 0.1896925
```

```r
# p drive_wheels
anova(lm.out3, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,10)]))$Pr[2]
```

```
## [1] 0.09223627
```

```r
# engine_type
anova(lm.out3, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,12)]))$Pr[2]
```

```
## [1] 0.0122903
```

```r
# fuel_system
anova(lm.out3, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,13)]))$Pr[2]
```

```
## [1] 2.843111e-08
```

```r
# aspiration
anova(lm.out3, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,14)]))$Pr[2]
```

```
## [1] 0.052273
```

Based on the p-values in this first backward elimination round, the highest p-value is 0.856 for `bore`. Let's remove it from the model, rerun the regression, and get the new p-values.

```r
# Model 4, now remove `bore` index 16

lm.out4 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16)])
round(summary(lm.out4)$coef[c(2:6, 26:30),4],3)
```

```
##         wheel_base            length             width            height
##              0.369             0.015             0.812             0.256
##        curb_weight       engine_size            stroke          peak_rpm
##              0.001             0.758             0.481             0.067
##         horsepower normalized_losses
##              0.439             0.126
```

```r
# p num_of_doors
anova(lm.out4, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,8)]))$Pr[2]
```

```
## [1] 0.7367373
```

```r
# p body_style
anova(lm.out4, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,9)]))$Pr[2]
```

```
## [1] 0.189434
```

```r
# p drive_wheels
anova(lm.out4, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,10)]))$Pr[2]
```

```
## [1] 0.08819382
```

```r
# engine_type
anova(lm.out4, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,12)]))$Pr[2]
```

```
## [1] 0.009908983
```

```r
# fuel_system
anova(lm.out4, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,13)]))$Pr[2]
```

```
## [1] 9.791969e-09
```

```r
# aspiration
anova(lm.out4, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,14)]))$Pr[2]
```

```
## [1] 0.05207403
```

The highest p-value now is 0.812 for `width` so we will remove it and continue backward elimination.

```r
# Model 5, now remove `width` index 5 as well.

lm.out5 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5)])
round(summary(lm.out5)$coef[c(2:5, 25:29),4],3)
```

```
##         wheel_base            length            height       curb_weight
##              0.383             0.014             0.219             0.001
##        engine_size            stroke          peak_rpm        horsepower
##              0.759             0.475             0.068             0.437
## normalized_losses
##              0.128
```

```r
# p num_of_doors
anova(lm.out5, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,8)]))$Pr[2]
```

```
## [1] 0.7447148
```

```r
# p body_style
anova(lm.out5, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,9)]))$Pr[2]
```

```
## [1] 0.1903532
```

```r
# p drive_wheels
anova(lm.out5, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,10)]))$Pr[2]
```

```
## [1] 0.08091571
```

```r
# engine_type
anova(lm.out5, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,12)]))$Pr[2]
```

```
## [1] 0.008662404
```

```r
# fuel_system
anova(lm.out5, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,13)]))$Pr[2]
```

```
## [1] 3.836538e-09
```

```r
# aspiration
anova(lm.out5, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,14)]))$Pr[2]
```

```
## [1] 0.05314911
```

The highest p-value now is 0.759 for `engine_size` so we will remove it and continue backward elimination.

```r
# Model 6, now remove `engine_size` index 15 as well.

lm.out6 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15)])
round(summary(lm.out6)$coef[c(2:5, 25:28),4],3)
```

```
##        wheel_base            length            height       curb_weight
##             0.411             0.011             0.186             0.000
##            stroke          peak_rpm        horsepower normalized_losses
##             0.503             0.029             0.442             0.129
```

```r
# p num_of_doors
anova(lm.out6, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8)]))$Pr[2]
```

```
## [1] 0.7833659
```

```r
# p body_style
anova(lm.out6, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,9)]))$Pr[2]
```

```
## [1] 0.1943348
```

```r
# p drive_wheels
anova(lm.out6, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,10)]))$Pr[2]
```

```
## [1] 0.07338889
```

```r
# engine_type
anova(lm.out6, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,12)]))$Pr[2]
```

```
## [1] 0.00242077
```

```r
# fuel_system
anova(lm.out6, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,13)]))$Pr[2]
```

```
## [1] 8.776801e-10
```

```r
# aspiration
anova(lm.out6, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,14)]))$Pr[2]
```

```
## [1] 0.007226591
```

The highest p-value now is 0.7834 for `num_of_doors` so we will remove it and continue backward elimination.

```r
# Model 7, now remove `num_of_doors` index 8 as well.

lm.out7 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8)])
round(summary(lm.out7)$coef[c(2:5, 24:27),4],3)
```

```
##        wheel_base            length            height       curb_weight
##             0.414             0.011             0.180             0.000
##            stroke          peak_rpm        horsepower normalized_losses
##             0.510             0.029             0.412             0.102
```

```r
# p body_style
anova(lm.out7, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,9)]))$Pr[2]
```

```
## [1] 0.09451036
```

```r
# p drive_wheels
anova(lm.out7, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,10)]))$Pr[2]
```

```
## [1] 0.06965504
```

```r
# engine_type
anova(lm.out7, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,12)]))$Pr[2]
```

```
## [1] 0.002243186
```

```r
# fuel_system
anova(lm.out7, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,13)]))$Pr[2]
```

```
## [1] 7.646698e-10
```

```r
# aspiration
anova(lm.out7, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,14)]))$Pr[2]
```

```
## [1] 0.007273297
```

The highest p-value now is 0.510 for `stroke` so we will remove it and continue backward elimination.

```r
# Model 8, now remove `stroke` index 17 as well.

lm.out8 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17)])
round(summary(lm.out8)$coef[c(2:5, 24:26),4],3)
```

```
##        wheel_base            length            height       curb_weight
##             0.358             0.013             0.200             0.000
##          peak_rpm        horsepower normalized_losses
##             0.034             0.339             0.091
```

```r
# p body_style
anova(lm.out8, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,9)]))$Pr[2]
```

```
## [1] 0.1003151
```

```r
# p drive_wheels
anova(lm.out8, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,10)]))$Pr[2]
```

```
## [1] 0.0771891
```

```r
# engine_type
anova(lm.out8, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,12)]))$Pr[2]
```

```
## [1] 0.00212153
```

```r
# fuel_system
anova(lm.out8, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,13)]))$Pr[2]
```

```
## [1] 7.722408e-10
```

```r
# aspiration
anova(lm.out8, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,14)]))$Pr[2]
```

```
## [1] 0.008359248
```

The highest p-value now is 0.358 for `wheel_base` so we will remove it and continue backward elimination.

```r
# Model 9, now remove `wheel_base` index 3 as well.

lm.out9 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3)])
round(summary(lm.out9)$coef[c(2:4, 23:25),4],3)
```

```
##            length            height       curb_weight          peak_rpm
##             0.002             0.135             0.000             0.033
##        horsepower normalized_losses
##             0.496             0.117
```

```r
# p body_style
anova(lm.out9, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,9)]))$Pr[2]
```

```
## [1] 0.1372446
```

```r
# p drive_wheels
anova(lm.out9, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,10)]))$Pr[2]
```

```
## [1] 0.1022763
```

```r
# engine_type
anova(lm.out9, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,12)]))$Pr[2]
```

```
## [1] 0.002884921
```

```r
# fuel_system
anova(lm.out9, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,13)]))$Pr[2]
```

```
## [1] 3.458528e-10
```

```r
# aspiration
anova(lm.out9, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,14)]))$Pr[2]
```

```
## [1] 0.006041898
```

The highest p-value now is 0.496 for `horsepower` so we will remove it and continue backward elimination.

```r
# Model 10, now remove `horsepower` index 20 as well.

lm.out10 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20)])
round(summary(lm.out10)$coef[c(2:4, 23:24),4],3)
```

```
##            length            height       curb_weight          peak_rpm
##             0.003             0.156             0.000             0.012
```

```
## normalized_losses
##             0.123
```

```r
# p body_style
anova(lm.out10, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,9)]))$Pr[2]
```

```
## [1] 0.1330392
```

```r
# p drive_wheels
anova(lm.out10, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,10)]))$Pr[2]
```

```
## [1] 0.124813
```

```r
# engine_type
anova(lm.out10, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,12)]))$Pr[2]
```

```
## [1] 0.001759056
```

```r
# fuel_system
anova(lm.out10, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,13)]))$Pr[2]
```

```
## [1] 4.24942e-14
```

```r
# aspiration
anova(lm.out10, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,14)]))$Pr[2]
```

```
## [1] 0.001653255
```

The highest p-value now is 0.156 for `height` so we will remove it and continue backward elimination.

```r
# Model 11, now remove `height` index 6 as well.

lm.out11 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6)])
round(summary(lm.out11)$coef[c(2:3, 22:23),4],3)
```

```
##            length       curb_weight          peak_rpm normalized_losses
##             0.000             0.000             0.014             0.249
```

```r
# p body_style
anova(lm.out11, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,9)]))$Pr[2]
```

```
## [1] 0.2089868
```

```r
# p drive_wheels
anova(lm.out11, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,10)]))$Pr[2]
```

```
## [1] 0.06808152
```

```r
# engine_type
anova(lm.out11, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,12)]))$Pr[2]
```

```
## [1] 0.003876086
```

```r
# fuel_system
anova(lm.out11, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,13)]))$Pr[2]
```

```
## [1] 7.933873e-14
```

```r
# aspiration
anova(lm.out11, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,14)]))$Pr[2]
```

```
## [1] 0.001494273
```

The highest p-value now is 0.249 for `normalized_losses` so we will remove it and continue backward elimination.

```
# Model 12, now remove `normalized_losses` index 21 as well.

lm.out12 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21)])
round(summary(lm.out12)$coef[c(2:3, 22),4],3)
```

```
##      length curb_weight    peak_rpm
##       0.000       0.000       0.004
```

```
# p body_style
anova(lm.out12, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9)]))$Pr[2]
```

```
## [1] 0.1762195
```

```
# p drive_wheels
anova(lm.out12, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,10)]))$Pr[2]
```

```
## [1] 0.05078986
```

```
# engine_type
anova(lm.out12, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,12)]))$Pr[2]
```

```
## [1] 0.005769344
```

```
# fuel_system
anova(lm.out12, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,13)]))$Pr[2]
```

```
## [1] 5.597311e-14
```

```
# aspiration
anova(lm.out12, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,14)]))$Pr[2]
```

```
## [1] 0.002061675
```

The highest p-value now is 0.176 for `body_style` so we will remove it and continue backward elimination.

```
# Model 13, now remove `body_style` index 9 as well.

lm.out13 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9)])
round(summary(lm.out13)$coef[c(2:3, 18),4],3)
```

```
##      length curb_weight    peak_rpm
##       0.006       0.000       0.010
```

```
# p drive_wheels
anova(lm.out13, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,10)]))$Pr[2]
```

```
## [1] 0.1131475
```

```
# engine_type
anova(lm.out13, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,12)]))$Pr[2]
```

```
## [1] 0.002604439
```

```
# fuel_system
anova(lm.out13, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,13)]))$Pr[2]
```

```
## [1] 3.365585e-16
```

```
# aspiration
anova(lm.out13, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,14)]))$Pr[2]
```

```
## [1] 0.002057631
```

The highest p-value now is 0.113 for `drive_wheels` so we will remove it and continue backward elimination.

```
# Model 14, now remove `drive_wheels` index 10 as well.

lm.out14 <- lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,10)])
round(summary(lm.out14)$coef[c(2:3, 16),4],3)
```

```
##      length curb_weight    peak_rpm
##       0.022       0.000       0.015
```

```
# engine_type
anova(lm.out14, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,10,12)]))$Pr[2]
```

```
## [1] 0.0003071912
```

```
# fuel_system
anova(lm.out14, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,10,13)]))$Pr[2]
```

```
## [1] 1.255773e-16
```

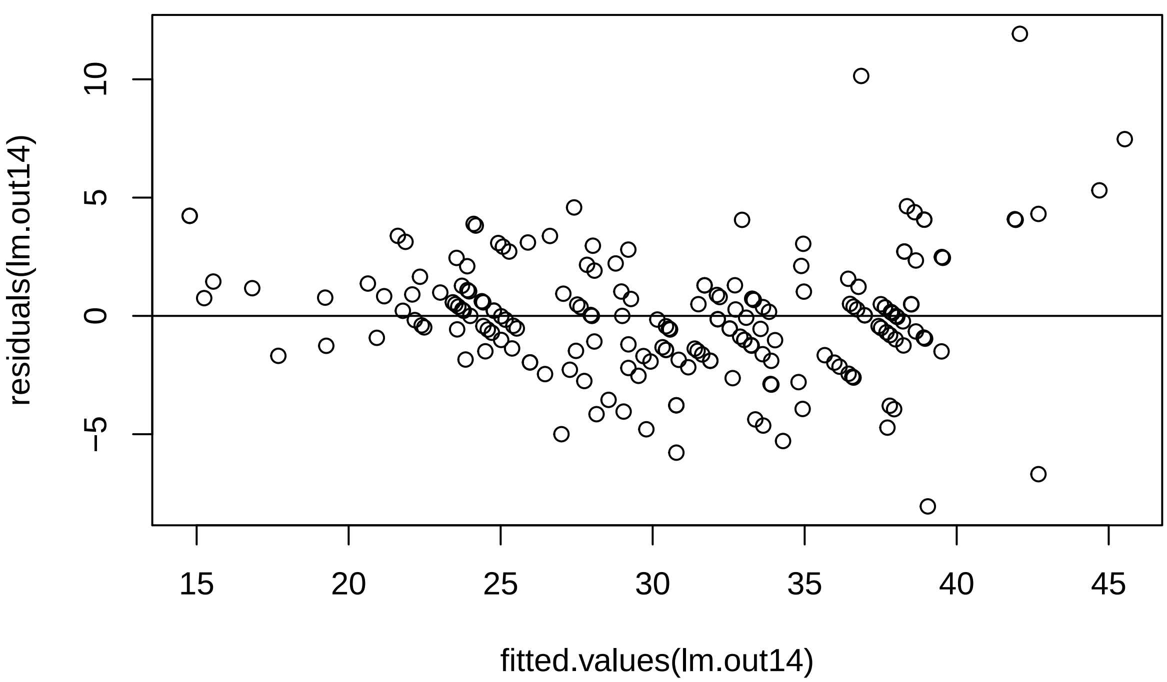
```
# aspiration
anova(lm.out14, lm(highway_mpg ~ ., data = df[,-c(1,2,11,18,16,5,15,8,17,3,20,6,21,9,10,14)]))$Pr[2]
```

```
## [1] 0.001025983
```

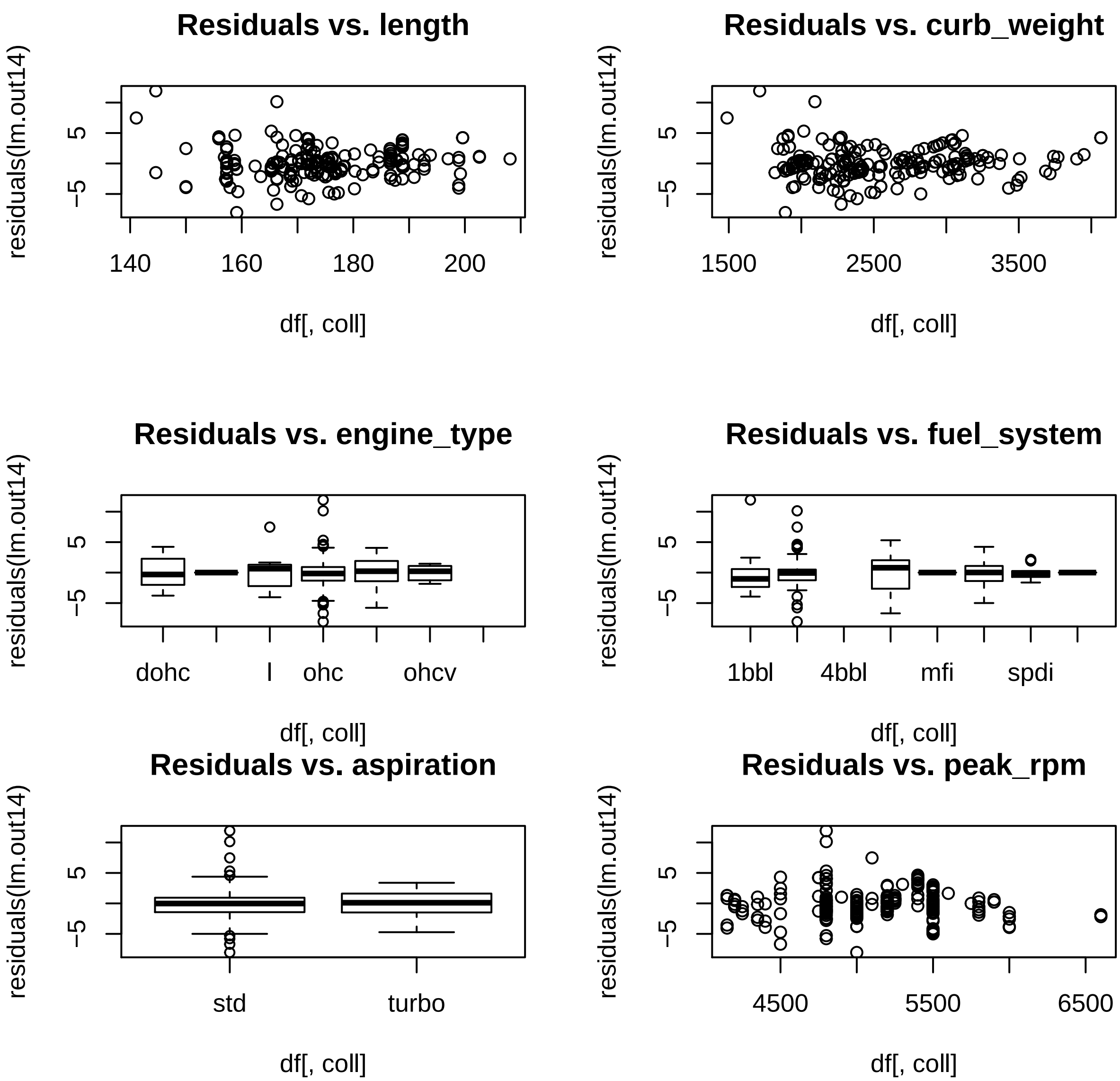
Recheck Diagnostics for Model 14

Finally, we have reached a model where all covariates are significant at the alpha = 0.10 level. We recheck the residual vs. fitted values plots to make sure no new issues were created in the model selection process.

```
plot(residuals(lm.out14) ~ fitted.values(lm.out14))
abline(h=0)
```



```r
par(mfrow=c(2,2))
for (coll in c(4, 7,12, 13, 14, 19)){
  plot(residuals(lm.out14) ~ df[,coll], main=paste("Residuals vs.", colnames(df)[coll]))
}
```



The residuals vs fitted values plot looks more evenly spread around the zero line for our reduced model than it did for our initial model (model 3) before backwards elimination. Also, the residual plots against each covariate no longer have some of the nonconstant variance that we saw earlier when we account for the amount of data at each part of the plots.

We will next consider including all two-way interactions for this model 14.

Check for Inclusion of Two-Way Interaction Terms

Consider the version of model 14 that now includes all pairwise (two-way) interaction terms between the six covariates. We denote this as model 14a. Note that there will be 6 choose 2 = 15 possible interaction terms to consider (potential multiple testing issue). We will use stepwise selection with AIC criteria to determine the best subset of all six covariates and their 15 interaction terms (not accounting for categorical covariates' dummy variables).

```r
lm.out14a <-lm(highway_mpg ~ (length + curb_weight + engine_type +
                              fuel_system + aspiration + peak_rpm)**2, data = df)

lm.out14b <-lm(highway_mpg ~ (length + curb_weight + engine_type +
                              fuel_system + aspiration + peak_rpm), data = df)

stepAIC(lm.out14b,direction="forward",
        scope=list(upper=lm.out14a,lower=lm.out14b), trace = 0)
```

```
##
## Call:
## lm(formula = highway_mpg ~ length + curb_weight + engine_type +
##     fuel_system + aspiration + peak_rpm + fuel_system:peak_rpm +
##     length:engine_type + length:fuel_system + engine_type:fuel_system +
##     aspiration:peak_rpm, data = df)
##
## Coefficients:
##                   (Intercept)                         length
##                     1.305e+02                      7.552e-03
##                   curb_weight               engine_typedohcv
##                    -9.039e-03                      7.927e+00
##                   engine_typel                 engine_typeohc
##                     4.289e+01                      1.290e+01
##                engine_typeohcf                engine_typeohcv
##                    -1.014e+01                      3.557e+01
##                fuel_system2bbl                 fuel_systemidi
##                    -6.777e+01                     -7.204e+01
##                 fuel_systemmfi                fuel_systemmpfi
##                    -1.271e+01                     -9.352e+01
##                fuel_systemspdi                fuel_systemspfi
##                    -1.127e+02                     -1.099e+01
##               aspirationturbo                       peak_rpm
##                    -1.439e+01                     -1.362e-02
##       fuel_system2bbl:peak_rpm        fuel_systemidi:peak_rpm
##                     1.422e-02                      1.634e-02
##        fuel_systemmfi:peak_rpm       fuel_systemmpfi:peak_rpm
##                            NA                      1.203e-02
##       fuel_systemspdi:peak_rpm       fuel_systemspfi:peak_rpm
##                     1.377e-02                             NA
##        length:engine_typedohcv             length:engine_typel
##                            NA                     -2.297e-01
##          length:engine_typeohc          length:engine_typeohcf
##                    -6.906e-02                      6.574e-02
##         length:engine_typeohcv          length:fuel_system2bbl
##                    -1.982e-01                     -9.424e-02
##          length:fuel_systemidi           length:fuel_systemmfi
##                    -5.244e-02                             NA
##         length:fuel_systemmpfi          length:fuel_systemspdi
##                     1.212e-01                      1.833e-01
##         length:fuel_systemspfi engine_typedohcv:fuel_system2bbl
##                            NA                             NA
##     engine_typel:fuel_system2bbl   engine_typeohc:fuel_system2bbl
```

```
##                            2.398e+00                            2.509e+00
##   engine_typeohcf:fuel_system2bbl   engine_typeohcv:fuel_system2bbl
##                                  NA                                   NA
##   engine_typedohcv:fuel_systemidi       engine_typel:fuel_systemidi
##                                  NA                            4.343e+00
##     engine_typeohc:fuel_systemidi    engine_typeohcf:fuel_systemidi
##                                  NA                                   NA
##    engine_typeohcv:fuel_systemidi   engine_typedohcv:fuel_systemmfi
##                                  NA                                   NA
##      engine_typel:fuel_systemmfi     engine_typeohc:fuel_systemmfi
##                                  NA                                   NA
##    engine_typeohcf:fuel_systemmfi    engine_typeohcv:fuel_systemmfi
##                                  NA                                   NA
## engine_typedohcv:fuel_systemmpfi      engine_typel:fuel_systemmpfi
##                                  NA                                   NA
##    engine_typeohc:fuel_systemmpfi   engine_typeohcf:fuel_systemmpfi
##                                  NA                                   NA
##   engine_typeohcv:fuel_systemmpfi  engine_typedohcv:fuel_systemspdi
##                                  NA                                   NA
##     engine_typel:fuel_systemspdi    engine_typeohc:fuel_systemspdi
##                                  NA                                   NA
##   engine_typeohcf:fuel_systemspdi   engine_typeohcv:fuel_systemspdi
##                                  NA                                   NA
## engine_typedohcv:fuel_systemspfi      engine_typel:fuel_systemspfi
##                                  NA                                   NA
##    engine_typeohc:fuel_systemspfi   engine_typeohcf:fuel_systemspfi
##                                  NA                                   NA
##   engine_typeohcv:fuel_systemspfi         aspirationturbo:peak_rpm
##                                  NA                            2.370e-03
```

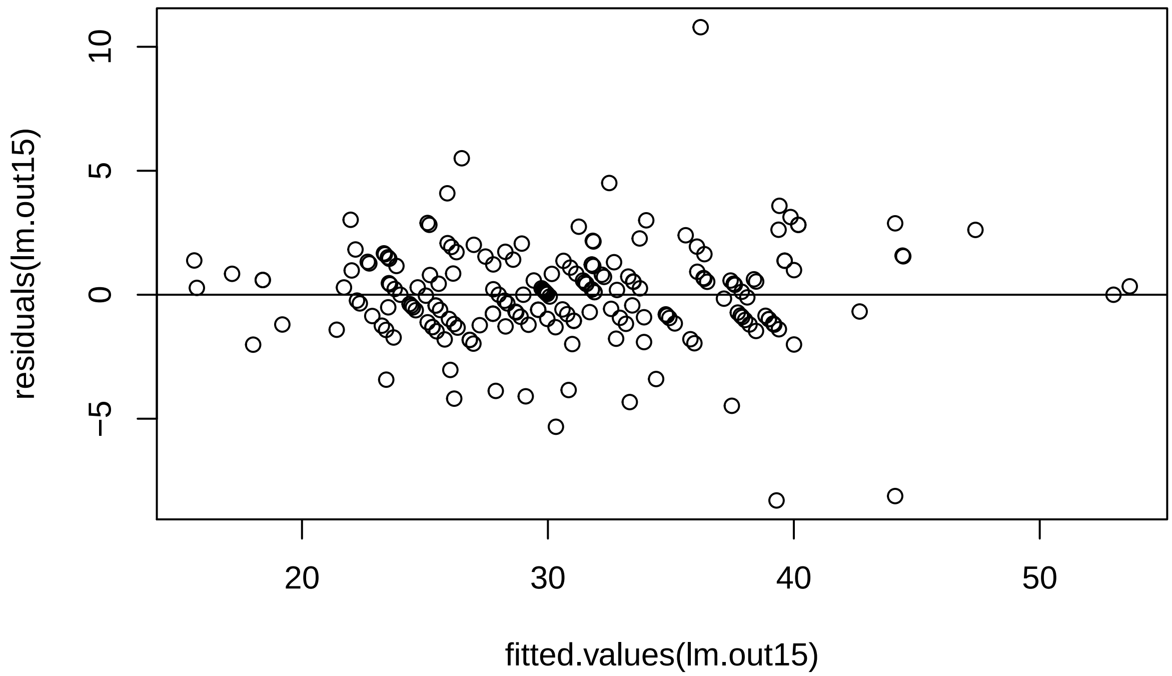
Based on forward stepwise selection using AIC criteria, we find that the best model subset with some pairwise interaction terms is the following:

```
lm.out15 <- lm(highway_mpg ~ length + curb_weight + engine_type +
               fuel_system + aspiration + peak_rpm + fuel_system:peak_rpm +
               length:engine_type + length:fuel_system + engine_type:fuel_system +
               aspiration:peak_rpm, data=df)
```

Recheck Diagnostics for Model 15 (best subset with two-way interactions)

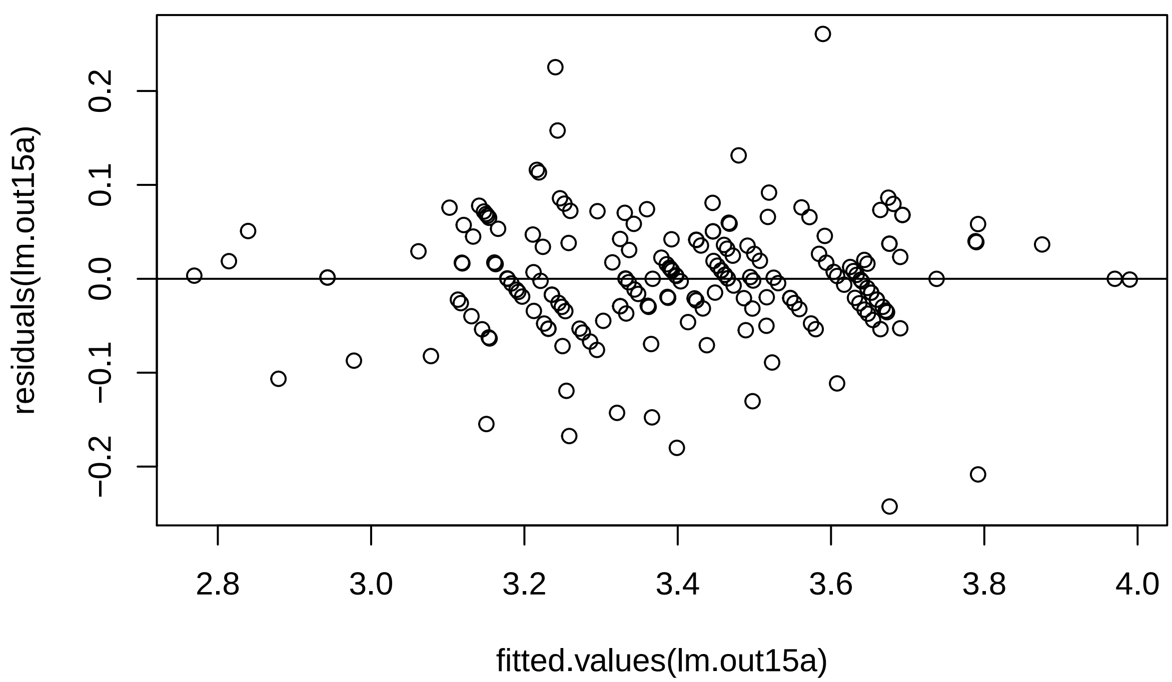
Let's recheck the model diagnostics for this final model to see if any adjustments still need to be made to satisfy assumptions.

```
plot(residuals(lm.out15) ~ fitted.values(lm.out15))
abline(h=0)
```



It is worth noting that the magnitude of the residuals and fitted values are actually quite large, and it looks like most of the residual points are around lower fitted values and the distribution looks skewed right. Because of this, we simply transform the response and rerun our same model against `log(highway_mpg)` now.

```
lm.out15a <- lm(log(highway_mpg) ~ length + curb_weight + engine_type +
                fuel_system + aspiration + peak_rpm + fuel_system:peak_rpm +
                length:engine_type + length:fuel_system + engine_type:fuel_system +
                aspiration:peak_rpm, data=df)
plot(residuals(lm.out15a) ~ fitted.values(lm.out15a))
abline(h=0)
```



After rerunning the model with the log-transformed response, we see a much nicer, more symmetric (less skewed) plot of the residuals vs. fitted values, and see that the points we were originally worried about as being potentially high leverage or outlier points no longer appear to have very different residuals at all from the majority of fitted values. This new residual plot also has much clearer constant variance than the residual plot of the older models, so the nonconstant variance problem is essentially resolved as well.

Conclusion

Since all of the model diagnostics look good on our final model, after performing backward elimination for variable selection and forward stepwise selection of interaction terms, we reach the best model for `highway_mpg`:

```
log(highway_mpg) ~ length + curb_weight + engine_type + fuel_system + aspiration + peak_rpm
+ fuel_system:peak_rpm + length:engine_type + length:fuel_system + engine_type:fuel_system
+ aspiration:peak_rpm
```