STAT34300 Final Project

Mingyu Liu & Huanqing Wang

2021/12/10

1. Introduction

In this report, we aim to use the `auto` data set to build a linear model for the response `highway_mpg` – the car's performance in miles per gallon during highway driving with all the other variables as potential covariates, and to draw appropriate conclusions regarding the associations of the response variable and the other covariates. The raw data set consists of 205 data points associated with 22 variables. Among the 22 variables, 9 of them are categorical, and the remaining 13 are numerical.

There are three main challenges facing this investigation. First of all, there is a large number of missing values contained in our data set, dropping which will throw away useful information that allows more precise inference. Carefully designed imputation strategies needs to be implemented in order to address the missing value problem appropriately. Secondly, the number of data points (hence the total degrees of freedom available) is relatively small compared to the number of covariate levels. This is particularly challenging if we want to build large models taking the effects of interactions into account. Proper model selection procedures need to be designed to perform model selection correctly. Finally, there exists exact collinearity between levels of categorical covariates, making estimation and inference of coefficients challenging. Ridge/LASSO regression methods will be necessary in order for the coefficients to be accurately estimated.

The report is structured as follows. In Section 2, we will pre-process the data and carry out imputation. Next, in Section 3, we will investigate the structure of all the possible covariates and check for outliers and high leverage points. Then, in Section 4, we will perform model selection, including searching for significant covariates and interaction effects. We will discuss model interpretation in Section 5 before conclusion in Section 6.

2. Data Pre-Processing

In this section, we load and pre-process the data. In particular, we will implement imputation strategies to deal with missing values in the data set. We first load the `auto` data set and convert all the "?" entries to `NA`.

```
library(faraway)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-3
```

```
library(Matrix)
library(MASS)
auto = read.csv(
  "./auto.txt",
  sep = "",
  na.strings = "?",
  stringsAsFactors = TRUE
)
```

Among the covariates with missing values, `num_of_doors` is categorical. Performing imputation on categorical covariates is challenging – regression-based methods are usually not well-suited for this type of problem, different imputation strategies are usually needed depending on whether the covariate is nominal or ordinal. The `amelia` package provides an easy-to-use multiple imputation method. However, this is unsuccessful in our data set due to exact collinearity among levels of other categorical covariates, which makes the data matrix singular.

```
print(nrow(auto))
```

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

```
print(colSums(is.na(auto))[which(colSums(is.na(auto)) > 0)])
```

```
##      num_of_doors              bore            stroke          peak_rpm
##                 2                 4                 4                 2
##        horsepower normalized_losses
##                 2                41
```

Since the variable `num_of_doors` only has two missing values, we simply remove the two points with `num_of_doors` missing, given the difficulty of imputing categorical variables. Removing the two points will introduce minimal bias into our regression analysis. With little prior knowledge about the cause of missing values, we assume missing values occur completely at random. We decide to perform imputation with regression for the remaining covariates.

A choice of the order of imputation also needs to be made. `normalized_losses` has the largest number of 41 missing values more than any other variables, and we decide to impute it last. All other covariates with missing values have exactly two missing points, so the order is less important. We decide to follow the column order: `bore` → `stroke` → `peak_rpm` → `horsepower` and lastly `normalized_losses`.

```
auto = auto[!is.na(auto$num_of_doors),]
miss_row = which(rowSums(is.na(auto[,-21])) > 0)
miss_col = which(colSums(is.na(auto)) > 0)
auto[miss_row, miss_col]
```

```
##     bore stroke peak_rpm horsepower normalized_losses
## 56    NA     NA     6000        101               150
## 57    NA     NA     6000        101               150
## 58    NA     NA     6000        101               150
## 59    NA     NA     6000        135               150
## 131 3.46    3.9       NA         NA                NA
## 132 3.46    3.9       NA         NA                NA
```

When imputing each column with regression, we fill the missing entries in the other columns not yet imputed by the mean. We first impute `bore`:

```
i = miss_col[1]
temp = auto
```

```
for(j in setdiff(miss_col, i)) {
  temp[is.na(temp[, j]), j] = mean(temp[, j], na.rm = T)
}
```

We regress the `bore` column with all the other covariate columns. Noticed that some missing data points have `num_of_cylinders=2` or `engine_type=rotor` or `fuel_system=4bbl` unique to the data point (all the other data points do not have this level). So, we omit these variables in the regression.

```
to_omit = which(colnames(auto) %in% c("engine_type", "num_of_cylinders", "fuel_system"))
model_temp = lm(as.formula(paste0(names(auto)[i], "~.")), data = temp[, -c(to_omit, ncol(temp))])
impute_row = which(is.na(auto[, i]))
pred = predict.lm(model_temp, temp[impute_row, -c(i, 22, to_omit)], interval = "confidence")[, 1]
auto[impute_row, i] = pred
```

Impute `stroke`:

```
i = miss_col[2]
temp = auto
for (j in setdiff(miss_col, i)) {
  temp[is.na(temp[, j]), j] = mean(temp[, j], na.rm = T)
}
```

Again, some missing values have `num_of_cylinders=2` or `engine_type=rotor` or `fuel_system=4bbl` levels that are unique to the data points. So, we omit these variables in the regression.

```
to_omit = which(colnames(auto) %in% c("engine_type", "num_of_cylinders", "fuel_system"))
model_temp = lm(as.formula(paste0(names(auto)[i], "~.")), data = temp[,-c(ncol(temp), to_omit)])
impute_row = which(is.na(auto[, i]))
pred = predict.lm(model_temp, temp[impute_row,-c(i, 22, to_omit)], interval = "confidence")[, 1]
auto[impute_row, i] = pred
```

Impute `peak_rpm`:

```
i = miss_col[3]
temp = auto
for (j in setdiff(miss_col, i)) {
  temp[is.na(temp[, j]), j] = mean(temp[, j], na.rm = T)
}
```

Similarly, we noticed that some missing values uniquely have `make=renault`, and all the other data points used in the model does not have this level. So, we omit `make`in the regression.

```
to_omit = which(colnames(auto) == "make")
model_temp = lm(as.formula(paste0(names(auto)[i], "~.")), data = temp[, -c(ncol(temp), to_omit)])
impute_row = which(is.na(auto[, i]))
pred = predict.lm(model_temp, temp[impute_row, -c(i, 22, to_omit)], interval = "confidence")[, 1]
auto[impute_row, i] = pred
```

Impute `horsepower`:

```
i = miss_col[4]
temp = auto
for (j in setdiff(miss_col, i)) {
  temp[is.na(temp[, j]), j] = mean(temp[, j], na.rm = T)
}
```

We noticed that some missing values have `make=renault`, and all the other data points used in the model does not have this level. So, we omit `make` in the regression.

```
to_omit = which(colnames(auto) == "make")
model_temp = lm(as.formula(paste0(names(auto)[i], "~.")), data = temp[, -c(ncol(temp), to_omit)])
impute_row = which(is.na(auto[, i]))
pred = predict.lm(model_temp, temp[impute_row, -c(i, 22, to_omit)], interval = "confidence")[, 1]
auto[impute_row, i] = pred
```

Finally, impute `normalized_losses`:

```
i = miss_col[5]
temp = auto
for (j in setdiff(miss_col, i)) {
  temp[is.na(temp[, j]), j] = mean(temp[, j], na.rm = T)
}
```

```
to_omit = which(colnames(auto) %in% c("num_of_cylinders", "make", "engine_type", "fuel_system"))
model_temp = lm(as.formula(paste0(names(auto)[i], "~.")), data = temp[, -c(ncol(temp), to_omit)])
impute_row = which(is.na(auto[, i]))
pred = predict.lm(model_temp, temp[impute_row, -c(i, 22, to_omit)], interval = "confidence")[, 1]
auto[impute_row, i] = pred
```

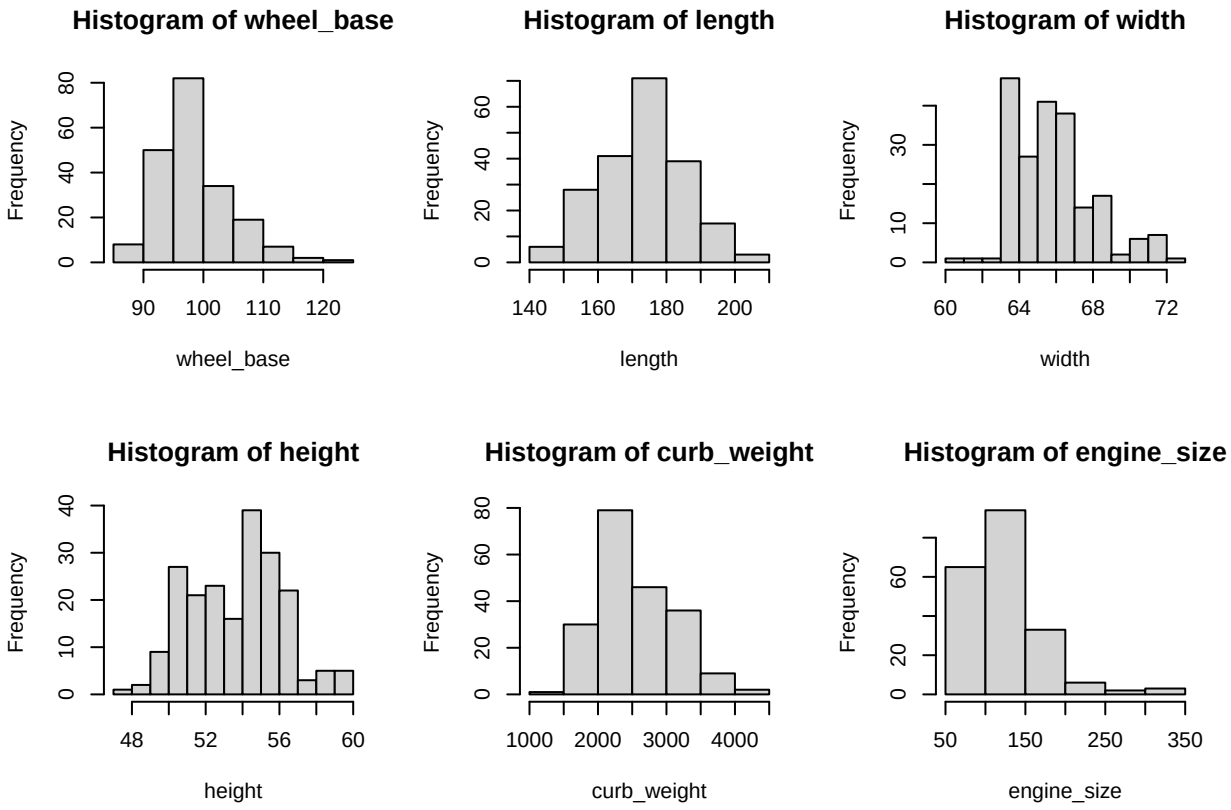
3. Data Exploration and Diagnostics

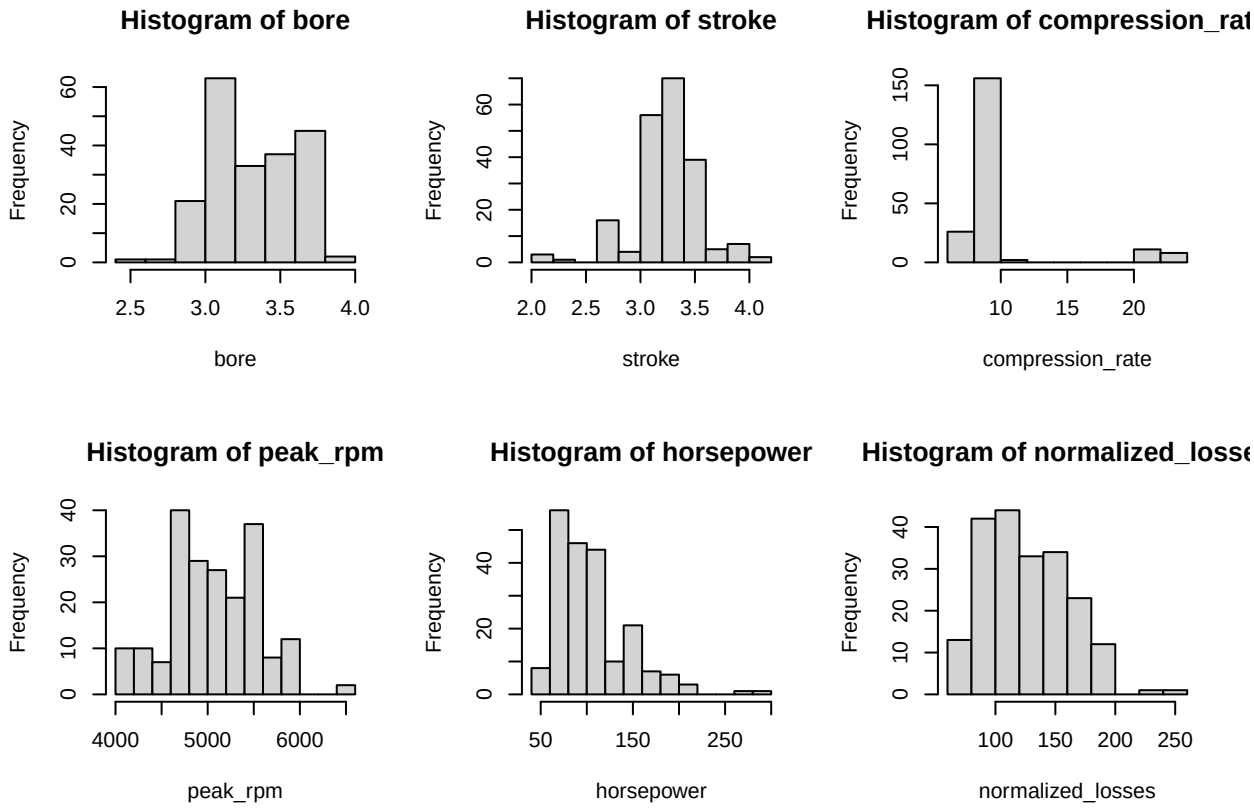
In this section, we (a) perform an exploratory analysis on our data set and (b) perform preliminary diagnostics, identify and remove outliers that will potentially cause problems in our later analysis.

3.1 Data Exploration

First, we plot the histogram for each numerical variable. We can see that most distributions are unimodal and not skewed. However, there are some extremely high values of `compression_rate` and `horsepower`, which might be causes of concern in later analysis.

```
par(mfrow = c(2, 3))
num_col = c()
for (i in 1:(ncol(auto) - 1)) {
  if (class(auto[, i]) != "factor") {
    num_col = c(num_col, i)
    hist(auto[, i], main  = paste0("Histogram of ", colnames(auto)[i]),xlab = colnames(auto)[i])
  }
}
```


Histogram of wheel_base
Frequency
0
20
40
60
80
90
100
110
120
wheel_base
Histogram of length
Frequency
0
20
40
60
140
160
180
200
length
Histogram of width
Frequency
0
10
30
60
64
68
72
width
Histogram of height
Frequency
0
10
20
30
40
48
52
56
60
height
Histogram of curb_weight
Frequency
0
20
40
60
80
1000
2000
3000
4000
curb_weight
Histogram of engine_size
Frequency
0
20
60
50
150
250
350
engine_size

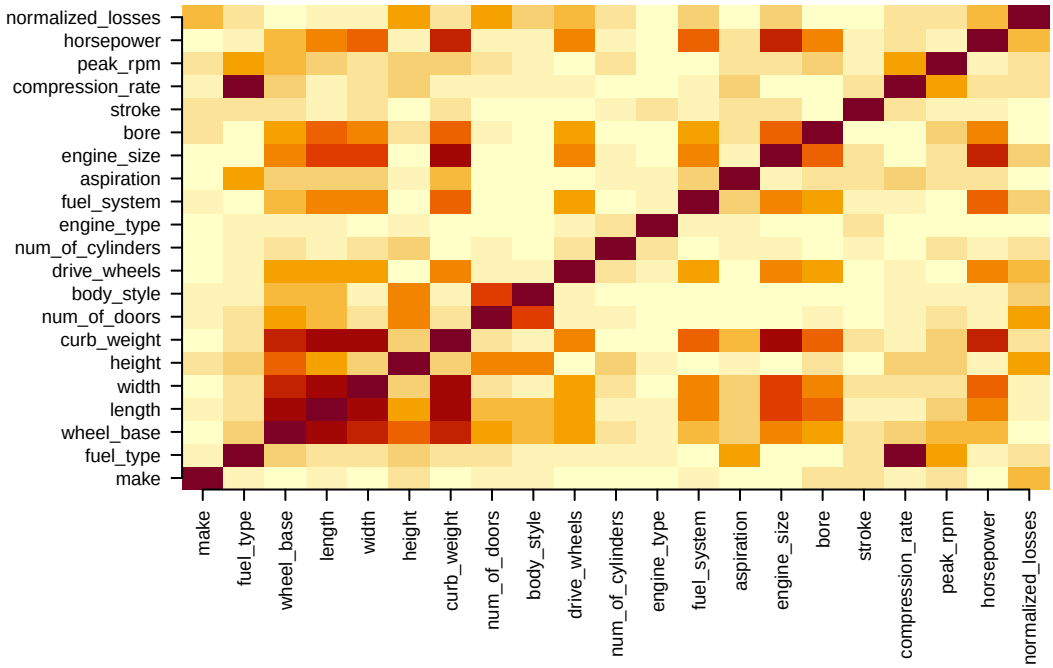



```
factor_col = setdiff(1:(ncol(auto) - 1), num_col)
```

For categorical variables, converting them into integers will not make much sense unless they are ordinal. Nevertheless, we still check their correlations since some variables seem to have ordinal structures.

```
temp = auto[, -ncol(auto)]
for (i in factor_col) {
  temp[, i] = as.numeric(auto[, i])
}
cor_matrix = cor(temp)
K = nrow(cor_matrix)
par(cex = 0.55, mar = c(9, 9, 9, 9))
image(
  1:K,
  1:K,
  abs(cor_matrix),
  xlab = '',
  ylab = '',
  axes = FALSE
)
axis(
  side = 1,
  at = 1:K,
  labels = colnames(cor_matrix),
  las = 3
```

```
)
axis(
  side = 2,
  at = 1:K,
  labels = colnames(cor_matrix),
  las = 2
)
```




```
colSums(abs(cor_matrix) > 0.8)
```

```
##              make         fuel_type        wheel_base            length
##                 1                 2                 2                 4
##             width            height       curb_weight      num_of_doors
##                 3                 1                 4                 1
##        body_style      drive_wheels  num_of_cylinders       engine_type
##                 1                 1                 1                 1
##       fuel_system        aspiration       engine_size              bore
##                 1                 1                 3                 1
##            stroke  compression_rate          peak_rpm        horsepower
##                 1                 2                 1                 2
## normalized_losses
##                 1
```

We can see that most covariates are only weakly correlated, but high correlations above 0.80 do exist. In particular, `curb_weight` has the largest number of highly correlated covariates, which we should keep an eye on in the later analysis. `coompression_rate` and `fuel_type` has the highest (negative) correlation -0.98.

3.2 Preliminary Diagnostics

In this section, we perform preliminary diagnostics to further explore the structure of the data. In particular, we will carry outlier analysis and remove points of concern from the data for further analysis. First of all, we fit a model of `highway_mpg` against all other covariates without considering any interaction.

```
model_out = lm(highway_mpg ~ ., data = auto)
```

From the summary table in Appendix 1, we can see that the categorical covariates `engine_type` and `fuel_system` have some `NA` in their estimates – an indication of some levels being linear combinations of the other covariates (i.e. the design matrix of this model does not have full rank). We investigate on this issue more closely.

```
X_temp=model.matrix(model_out)
rank_removed=sapply(1:ncol(X_temp),function(x){rankMatrix(X_temp[,-x])})
dependentcol=which(rank_removed==rankMatrix(X_temp))
colnames(X_temp)[dependentcol]
```

```
## [1] "(Intercept)"           "makepeugot"            "fuel_typegas"
## [4] "num_of_cylindersthree" "num_of_cylinderstwo"   "engine_typel"
## [7] "engine_typerotor"      "fuel_systemidi"
```

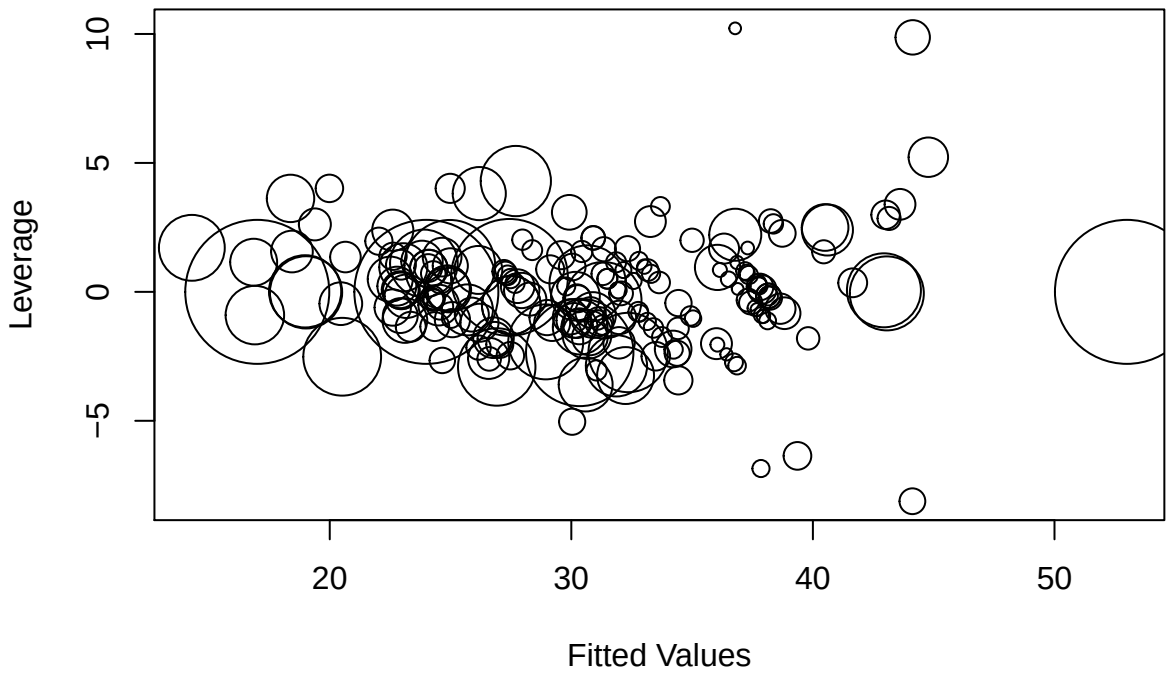
If a column of the design matrix is a linear combination of the other columns, then deleting that column from the matrix will not affect the matrix rank. Based on the results shown we can conclude that the covariates `engine_type` and `fuel_system` indeed have levels collinear with the other levels in our design matrix. This will cause problems in the following leverage point diagnostic and studentized residuals. Hence, we temporarily omit these two variables for the purpose of diagnostics.

```
to_omit = which(colnames(auto) %in% c("engine_type", "fuel_system"))
model_out1 = lm(highway_mpg ~ ., data = auto[, -to_omit])
```

Leverage

First, we check leverage points. There are indeed several points with high leverage. In particular, two points with low fitted value and one point with the largest fitted value have unusually high leverage. This is expected because points far outside the "typical" X's are considered as leverage points.

```
X = model.matrix(model_out1)
H = (X %*% solve(t(X) %*% X, t(X)))
lev = diag(H)
plot(
  model_out1$fit,
  model_out1$res,
  cex = 10 * lev,
  xlab = "Fitted Values",
  ylab = "Leverage"
)
```



Studentized Residuals

Then, we calculate which point has significant studentized residuals at the 0.05 significance level, with Bonferroni correction.

```
model_out2 = lm(highway_mpg ~ ., data = data_temp[, -to_omit])
n = nrow(data_temp)
df = summary(model_out2)$df[1]
res = abs(studres(model_out2))
tval = qt(1 - 0.05 / 2 / n, df)
```

```
print(res[res > tval])
```

```
##       31      161
## 5.014299 4.675512
```

```
print(paste0('Bonferroni-adjusted threshold: ', tval))
```

```
## [1] "Bonferroni-adjusted threshold: 3.9655146397967"
```

Cook's Distance

The Cook's distance of a point $(\mathbf{x}_i, y_i)$ is a measure of influence given by the sum of the squares of the shift in fitted values when point i is removed. The Cook's distance for the ith data point is defined to be

$$C_i = \frac{(\hat{y} - \hat{y}_{-i})^T (\hat{y} - \hat{y}_{-i})}{ps^2} = \frac{r_i^2 h_{ii}}{p(1 - h_{ii})}$$

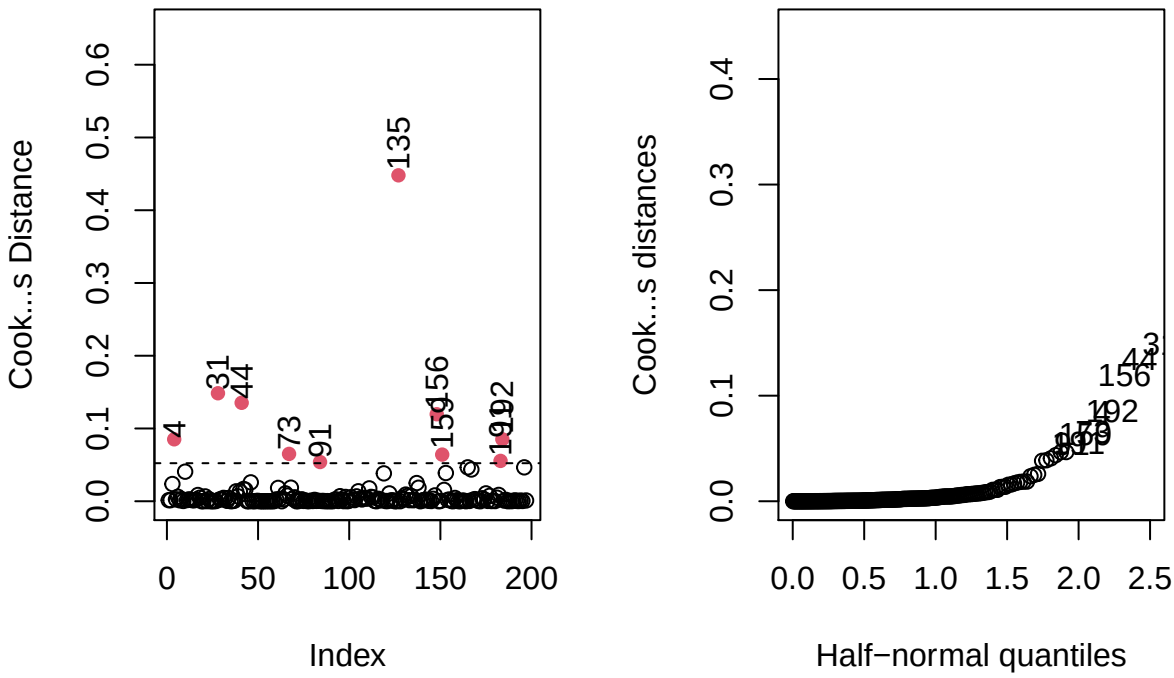
High influence occurs where there is a misfit and large leverage. A rough rule of thumb is that $|r_i| > 2$ and $h_{ii} > 2p/n$ are separate causes for concern. Substituting into the equation above, points cause problems of concern if Cook's distance exceeds

$$C_k \gtrsim \frac{8}{n - 2p}$$

```
cooks = cooks.distance(model_out2)
i = cooks > 8 / (nrow(data_temp) - 2 * (ncol(data_temp)))
print(sort(cooks[which(i)]))
```

```
##         91        191        159         73          4        192        156
## 0.05389349 0.05546276 0.06408808 0.06507328 0.08512382 0.08512382 0.11957615
##         44         31        135
## 0.13532027 0.14844811 0.44798306
```

```
par(mfrow = c(1, 2))
index = row.names(data_temp)
plot(
  cooks,
  ylim = c(0, 0.65),
  pch = 1 + 15 * i,
  col = 1 + i,
  ylab = "Cook's Distance"
)
text(
  x = which(i),
  y = cooks[which(i)],
  labels = index[which(i)],
  srt = 90,
  adj = -0.1
)
abline(8 / (n - 2 * ncol(auto)), 0, lty = 2)
halfnorm(
  cooks,
  nlab = length(which(i)),
  xlim = c(0, 2.5),
  labs = index,
  ylab = "Cook's distances"
)
```



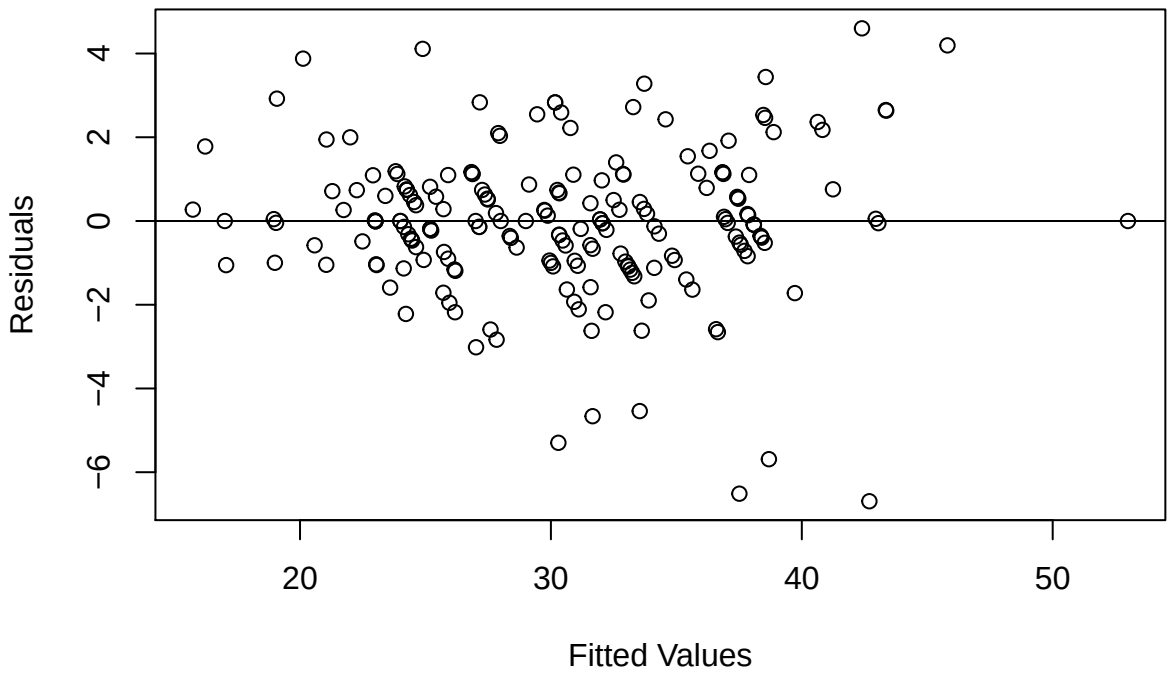
From the above plots, we can see that data point 135 has a significantly high Cook's distance, while the other points are below or near the boundary. We remove this data point along with the two points having significant studentized residual p-value.

```
delete = c(names(res)[res > tval], "135")
auto = auto[-which(row.names(auto) %in% delete), ]
```

Fitted Residual Plots

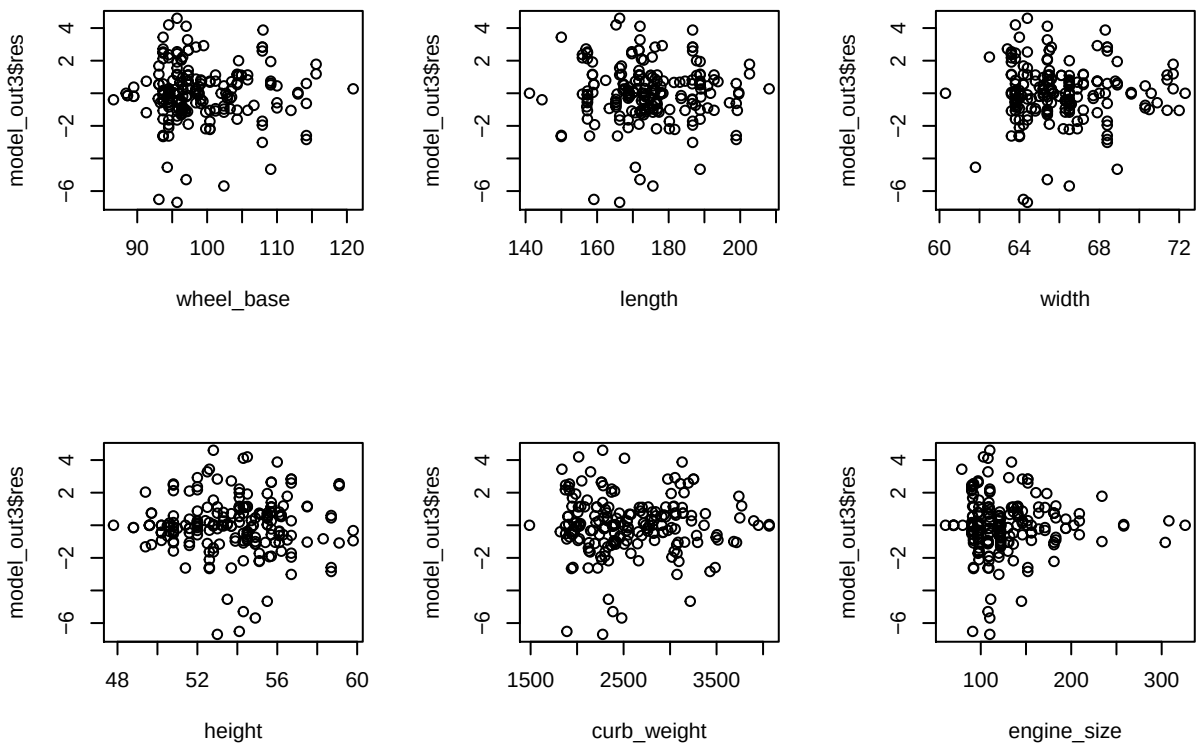
Using data with outliers removed, we fit a model of `highway_mpg` against all other covariates again without any interaction and look at the residual plot. From the residual plot, we can see that the constant variance assumption appears to be obeyed, but there are some extreme residuals around fitted `highway_mpg` equal to 40. Also, there seems to be a negative trend among adjacent points. The model violation suggests variations in response unexplained by our covariates – interactions between covariates will potentially be necessary.

```
model_out3 = lm(highway_mpg ~ ., data = auto)
plot(model_out3$fit,
     model_out3$res,
     xlab = "Fitted Values",
     ylab = "Residuals")
abline(h = 0)
```



Plot the residual for each covariate

```
# Appendix 3 for all plots
par(mfrow = c(2, 3))
for (i in num_col[1:6]) {
  plot(auto[, i], model_out3$res, xlab = colnames(auto)[i])
}
```

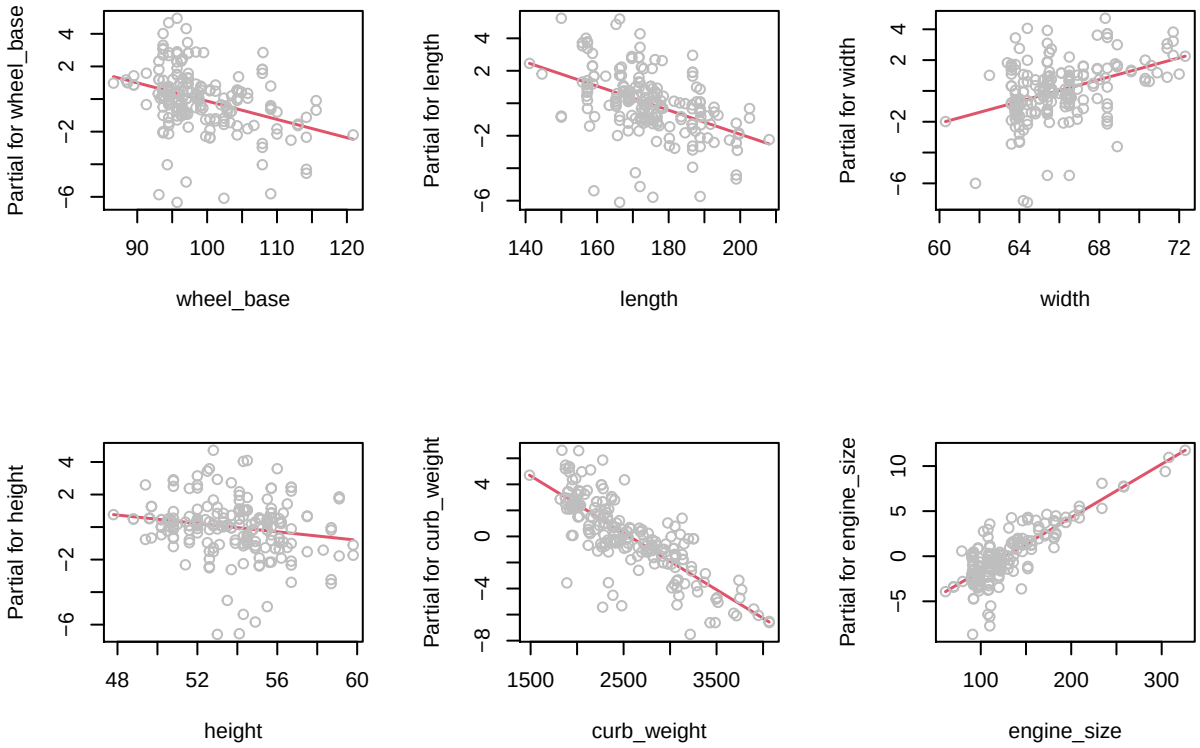


Observations & potential issues: * `num_of_cylinders` has most values concentrated at 4, many levels have only a single observation * `compression_rate` has two obvious clusters * Most plots show one to two outlying residuals * Possibly ordinal covariates: `num_of_doors`, `num_of_cylinders`, `drive_wheels`, `fuel_system` — treat as factor or quantitative?

Partial Residual Plots

We also plot the partial residuals individually for the numerical covariates. Partial residual plots look at the marginal relationship between the response and the predictor after the effect of the other predictors has been removed. The trends in the plots are consistent with the coefficients in the fitted model.

```
# Appendix 4 for all plots
par(mfrow = c(2, 3))
for (i in num_col[1:6]) {
  termplot(model_out3, partial.resid = TRUE, terms = i)
}
```



From the diagnostic plots, there are no obvious non-linear trends between the residuals / partial residuals against the covariates. In addition, the constant variance assumption seems to hold. Therefore, we decide not to transform the response or any variable.

4. Model Selection

4.1 Initial Regression

First of all, we fit a model of `highway_mpg` against all the other covariates without considering interactions. From the ANOVA table, the covariates that are significant at the 0.05 level are `make`, `fuel_type`, `wheel_base`, `length`, `width`, `height`, `curb_weight`, `num_of_cylinders`, `engine_type`, `fuel_system`, `aspiration`, `bore` and `compression_rate`.

```
model_1 = lm(highway_mpg ~ ., data=auto)
anova_1 = anova(model_1)
anova_1
```

```
## Analysis of Variance Table
##
## Response: highway_mpg
##                    Df Sum Sq Mean Sq  F value    Pr(>F)
## make               21 4025.7  191.70  46.1082 < 2.2e-16 ***
## fuel_type           1  916.7  916.69 220.4856 < 2.2e-16 ***
## wheel_base          1  521.8  521.76 125.4950 < 2.2e-16 ***
## length              1  877.6  877.61 211.0870 < 2.2e-16 ***
## width               1  140.8  140.83  33.8723 3.795e-08 ***
## height              1   61.5   61.51  14.7953 0.0001809 ***
## curb_weight         1 1013.5 1013.51 243.7746 < 2.2e-16 ***
## num_of_doors        1    9.0    9.00   2.1639 0.1435132
## body_style          4    3.2    0.80   0.1917 0.9424339
## drive_wheels        2   22.5   11.26   2.7081 0.0701328 .
## num_of_cylinders    6  207.0   34.50   8.2988 1.051e-07 ***
## engine_type         4   43.3   10.83   2.6038 0.0384683 *
## fuel_system         6  128.9   21.49   5.1682 7.836e-05 ***
## aspiration          1   79.3   79.32  19.0774 2.416e-05 ***
## engine_size         1   12.0   12.00   2.8851 0.0916071 .
## bore                1   23.3   23.35   5.6151 0.0191614 *
## stroke              1    3.6    3.61   0.8684 0.3529825
## compression_rate    1   29.8   29.83   7.1747 0.0082728 **
## peak_rpm            1   14.7   14.66   3.5263 0.0624672 .
## horsepower          1    2.3    2.32   0.5583 0.4561911
## normalized_losses   1    1.1    1.13   0.2706 0.6037216
## Residuals         141  586.2    4.16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We remove all insignificant covariates and refit the model – all covariates in the reduced model are now significant at the 0.05 threshold.

```
anova_1 = anova_1[1:(dim(anova_1)[1] - 1), ]
selected = rownames(anova_1[anova_1[, "Pr(>F)"] < 0.05, ])
excluded = rownames(anova_1[anova_1[, "Pr(>F)"] >= 0.05, ])
model_1a = lm(highway_mpg ~ ., data = auto[, c("highway_mpg", selected)])
anova_1a = anova(model_1a)
anova_1a
```

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

```
##
## Response: highway_mpg
##                    Df Sum Sq Mean Sq  F value    Pr(>F)
## make               21 4025.7  191.70  45.6705 < 2.2e-16 ***
## fuel_type           1  916.7  916.69 218.3928 < 2.2e-16 ***
## wheel_base          1  521.8  521.76 124.3038 < 2.2e-16 ***
## length              1  877.6  877.61 209.0834 < 2.2e-16 ***
## width               1  140.8  140.83  33.5508 3.823e-08 ***
## height              1   61.5   61.51  14.6548 0.0001878 ***
## curb_weight         1 1013.5 1013.51 241.4607 < 2.2e-16 ***
## num_of_cylinders    6  214.6   35.77   8.5229 5.421e-08 ***
## engine_type         4   49.5   12.38   2.9495 0.0220520 *
## fuel_system         6  123.0   20.50   4.8830 0.0001357 ***
## aspiration          1   76.2   76.20  18.1541 3.546e-05 ***
## bore                1   36.4   36.43   8.6803 0.0037202 **
## compression_rate    1   24.4   24.38   5.8092 0.0171291 *
## Residuals         153  642.2    4.20
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It should be pointed out that the F-statistics from ANOVA are highly dependent on the order of the covariates being included, and changing the order may affect our conclusion about the variable significance. To address this issue, we use F-test to compare model `model_1a` against models with one additional insignificant covariate added back, one at a time. From the result below, we verify that the excluded covariates are indeed insignificant. We will work with the smaller model from this point on.

```
for (exc in excluded) {
  md = lm(highway_mpg ~ ., data = auto[, c("highway_mpg", selected, exc)])
  F_stat = format(round(anova(model_1a, md)[, "Pr(>F)"][2], 3), nsmall = 3)
  cat(paste0("F-statistic=", F_stat),
      "with",
      exc,
      "added back",
      "\n")
}
```

```
## F-statistic=0.647 with num_of_doors added back
## F-statistic=0.799 with body_style added back
## F-statistic=0.143 with drive_wheels added back
## F-statistic=0.097 with engine_size added back
## F-statistic=0.928 with stroke added back
## F-statistic=0.067 with peak_rpm added back
## F-statistic=0.494 with horsepower added back
## F-statistic=0.264 with normalized_losses added back
```

4.2 Interactions

Next, we consider models that include two-way interactions between the selected covariates. It is tempting to include all possible two-way interactions between the selected the covariates and perform backward elimination to select the set of significant covariates. However, we note that the number of interactions overshoots the 158 degrees of freedom remaining – making estimation of the coefficients impossible. In addition, the large number of degrees of freedom will likely to cause huge multiple testing issues – many coefficients could

appear to be significant by random chance. To address the two issues, we carry out a forward selection procedure and adopt a more conservative significance threshold.

At each step, we progressively add one additional interaction into `model_1a` and evaluated the F-statistic compared with `model_1a`. We include an interaction if the F-statistic is below the 0.01 significance threshold.

```
forward = function (formula_init, model_init) {
  inter_li = c()
  F_stat_li = c()
  for (sel_i in selected) {
    for (sel_j in selected) {
      if (sel_i != sel_j) {
        inter = paste0(sel_i, ":", sel_j)
        formula_2_temp = paste0(formula_init, "+", inter)
        model_2_temp = lm(formula_2_temp, data = auto[, c("highway_mpg", selected)])
        F_stat = anova(model_init, model_2_temp)[, "Pr(>F)"][2]
        if (!is.na(F_stat)) {
          inter_li = c(inter_li, inter)
          F_stat_li = c(F_stat_li, F_stat)
        }
      }
    }
  }
  return(list(min(F_stat_li), inter_li[which.min(F_stat_li)]))
}
```

```
formula_1a = paste0("highway_mpg ~ ", paste0(selected, collapse = "+"))
output_2a = forward(formula_1a, model_1a)
F_stat_2a = output_2a[[1]]
inter_2a = output_2a[[2]]
cat("The most significant F-statistics:",
    F_stat_2a,
    "achieved with",
    inter_2a)
```

```
## The most significant F-statistics: 3.463269e-07 achieved with make:compression_rate
```

We notice that the degrees of freedom taken by the interaction `make:compression_rate` is 15, lower than the 21 degrees of freedom we anticipate. This is due to exact collinearity between the interaction term and `make` (the covariate `compression_rate` is numeric but only takes upon a discrete set of values). Some coefficients cannot be appropriately estimated. This problem will be addressed in Section 5 of the report. We repeat the forward selection until exhausting interaction terms significant at the 0.01 significance level.

```
formula_2a = paste0(formula_1a, "+", inter_2a)
model_2a = lm(formula_2a, data = auto[, c("highway_mpg", selected)])
anova(model_2a)
```

```
## Analysis of Variance Table
##
## Response: highway_mpg
##                       Df Sum Sq Mean Sq  F value    Pr(>F)
## make                  21 4025.7  191.70  62.1004 < 2.2e-16 ***
## fuel_type              1  916.7  916.69 296.9590 < 2.2e-16 ***
```

```
## wheel_base              1  521.8  521.76 169.0217 < 2.2e-16 ***
## length                  1  877.6  877.61 284.3006 < 2.2e-16 ***
## width                   1  140.8  140.83  45.6206 3.679e-10 ***
## height                  1   61.5   61.51  19.9269 1.658e-05 ***
## curb_weight             1 1013.5 1013.51 328.3255 < 2.2e-16 ***
## num_of_cylinders        6  214.6   35.77  11.5890 1.748e-10 ***
## engine_type             4   49.5   12.38   4.0105 0.0041457 **
## fuel_system             6  123.0   20.50   6.6397 3.457e-06 ***
## aspiration              1   76.2   76.20  24.6850 1.965e-06 ***
## bore                    1   36.4   36.43  11.8030 0.0007815 ***
## compression_rate        1   24.4   24.38   7.8991 0.0056659 **
## make:compression_rate  15  216.2   14.41   4.6694 3.463e-07 ***
## Residuals             138  426.0    3.09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
formula_step = formula_2a
model_step = model_2a
F_stat_step = F_stat_2a
inter_step = inter_2a

while (F_stat_step < 0.01) {
  output_step = forward(formula_step, model_step)
  F_stat_step = output_step[[1]]
  inter_step = output_step[[2]]
  if (F_stat_step >= 0.01) {
    break
  }
  formula_step = paste0(formula_step, "+", inter_step)
  cat("The most significant F-statistics:",
      F_stat_step,
      "achieved with",
      inter_step,
      "\n")
  model_step = lm(formula_step, data = auto[, c("highway_mpg", selected)])
}
```

```
## The most significant F-statistics: 0.0001584984 achieved with length:compression_rate
## The most significant F-statistics: 0.001060149 achieved with fuel_type:compression_rate
```

The significant interactions are `make:compression_rate`, `length:compression_rate` and `fuel_type:compression_rate`.

```
formula_2b = formula_step
model_2b = model_step
anova_2b = anova(model_2b)
anova_2b
```

```
## Analysis of Variance Table
##
## Response: highway_mpg
##                            Df Sum Sq Mean Sq  F value    Pr(>F)
## make                       21 4025.7  191.70  73.5369 < 2.2e-16 ***
## fuel_type                   1  916.7  916.69 351.6478 < 2.2e-16 ***
```

```
## wheel_base                    1  521.8  521.76 200.1492 < 2.2e-16 ***
## length                        1  877.6  877.61 336.6581 < 2.2e-16 ***
## width                         1  140.8  140.83  54.0222 1.667e-11 ***
## height                        1   61.5   61.51  23.5966 3.220e-06 ***
## curb_weight                   1 1013.5 1013.51 388.7908 < 2.2e-16 ***
## num_of_cylinders              6  214.6   35.77  13.7233 3.769e-12 ***
## engine_type                   4   49.5   12.38   4.7491 0.0012848 **
## fuel_system                   6  123.0   20.50   7.8625 2.796e-07 ***
## aspiration                    1   76.2   76.20  29.2310 2.801e-07 ***
## bore                          1   36.4   36.43  13.9767 0.0002717 ***
## compression_rate              1   24.4   24.38   9.3538 0.0026805 **
## make:compression_rate        15  216.2   14.41   5.5294 1.145e-08 ***
## length:compression_rate       1   42.3   42.28  16.2188 9.335e-05 ***
## fuel_type:compression_rate    1   29.2   29.19  11.1957 0.0010601 **
## Residuals                   136  354.5    2.61
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Adding Back Covariates

It is still possible that one or more of the covariates removed in `model_1a` might be significant once we allow for interactions, which we now test one at a time with the F-test. For each of the covariates removed in `model_1a`, we include its two-way interactions with the selected covariates and compare with the model `model_2b`.

```
for (exc in excluded) {
  formula_md = paste0(formula_2b, "+", exc, "*", "(", paste0(selected, collapse="+"), ")")
  md = lm(formula_md, data = auto[, c("highway_mpg", selected, exc)])
  F_stat = format(round(anova(model_2b, md)[, "Pr(>F)"][2], 3), nsmall=3)
  cat(paste0("F-statistic=", F_stat), "with", exc, "added back (allowing for interactions)", "\n")
}
```

```
## F-statistic=0.926 with num_of_doors added back (allowing for interactions)
## F-statistic=0.018 with body_style added back (allowing for interactions)
## F-statistic=0.018 with drive_wheels added back (allowing for interactions)
## F-statistic=0.862 with engine_size added back (allowing for interactions)
## F-statistic=0.223 with stroke added back (allowing for interactions)
## F-statistic=0.209 with peak_rpm added back (allowing for interactions)
## F-statistic=0.145 with horsepower added back (allowing for interactions)
## F-statistic=0.918 with normalized_losses added back (allowing for interactions)
```

We observe that `body_style` and `drive_wheels` now appear to be more significant than when the terms are included individually. Both of them are now significant at the 0.05 level, though not meeting our 0.01 threshold for including additional interaction terms. In particular, the significance of `drive_wheels` comes from its interaction with `curb_weight`.

```
formula_3 = paste0(formula_2b, "+", "drive_wheels", "*", "(", paste0(selected, collapse="+"), ")")
model_3 = lm(formula_3, data = auto[, c("highway_mpg", selected, "drive_wheels")])
anova_3 = anova(model_3)
anova_3
```

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

```
##
## Response: highway_mpg
##                                Df Sum Sq Mean Sq  F value    Pr(>F)
## make                           21 4025.7  191.70  84.3766 < 2.2e-16 ***
## fuel_type                       1  916.7  916.69 403.4821 < 2.2e-16 ***
## wheel_base                      1  521.8  521.76 229.6520 < 2.2e-16 ***
## length                          1  877.6  877.61 386.2829 < 2.2e-16 ***
## width                           1  140.8  140.83  61.9853 2.406e-12 ***
## height                          1   61.5   61.51  27.0749 8.942e-07 ***
## curb_weight                     1 1013.5 1013.51 446.1001 < 2.2e-16 ***
## num_of_cylinders                6  214.6   35.77  15.7462 4.793e-13 ***
## engine_type                     4   49.5   12.38   5.4492 0.0004793 ***
## fuel_system                     6  123.0   20.50   9.0215 4.864e-08 ***
## aspiration                      1   76.2   76.20  33.5398 6.492e-08 ***
## bore                            1   36.4   36.43  16.0369 0.0001121 ***
## compression_rate                1   24.4   24.38  10.7326 0.0014016 **
## drive_wheels                    2   16.4    8.18   3.5994 0.0305471 *
## make:compression_rate          15  229.1   15.28   6.7237 4.028e-10 ***
## length:compression_rate         1   41.9   41.93  18.4567 3.719e-05 ***
## fuel_type:compression_rate      1   30.9   30.87  13.5867 0.0003528 ***
## make:drive_wheels               5   15.7    3.15   1.3856 0.2350702
## fuel_type:drive_wheels          1    0.1    0.06   0.0267 0.8705472
## wheel_base:drive_wheels         2   13.4    6.72   2.9569 0.0560528 .
## length:drive_wheels             2    1.4    0.71   0.3131 0.7317863
## width:drive_wheels              2    2.0    1.00   0.4407 0.6446672
## height:drive_wheels             1    3.4    3.37   1.4815 0.2261056
## curb_weight:drive_wheels        2   21.9   10.96   4.8254 0.0097676 **
## num_of_cylinders:drive_wheels   1    0.7    0.66   0.2915 0.5903117
## engine_type:drive_wheels        1    0.2    0.16   0.0716 0.7895461
## fuel_system:drive_wheels        2    9.9    4.95   2.1795 0.1178664
## aspiration:drive_wheels         1    0.1    0.08   0.0369 0.8481077
## bore:drive_wheels               1    0.1    0.05   0.0220 0.8823057
## compression_rate:drive_wheels   1    0.6    0.64   0.2820 0.5964315
## Residuals                     112  254.5    2.27
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Degrees of Freedom

The `model_2b` now takes up 63 degrees of freedom (among the total 203 degrees of freedom available). We will not further consider higher order interactions to avoid having models with intractably high number of covariates, poor interpretability or potentially insufficient degrees of freedom for inference.

```
anova_2b = anova_2b[1:(dim(anova_2b)[1] - 1), ]
sum(anova_2b[, "Df"])
```

```
## [1] 63
```

5. Parameter Estimation

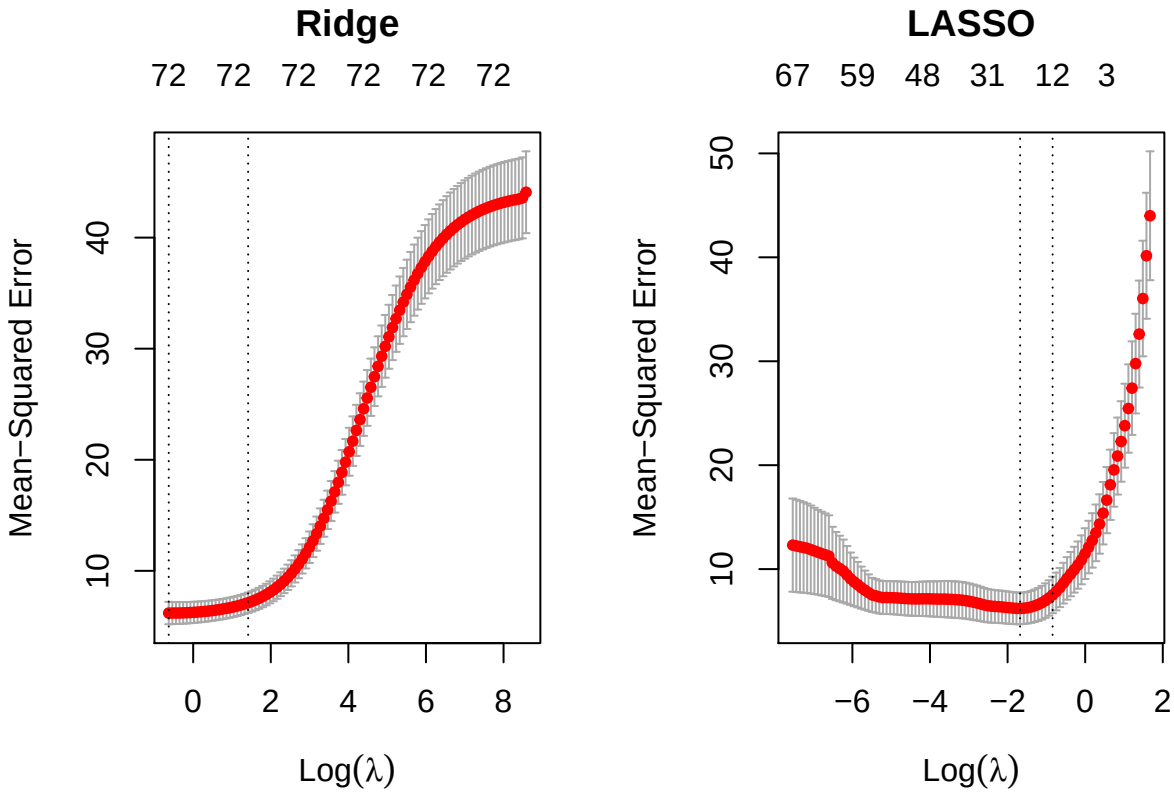
5.1 Elastic Net Regression

Having selected the group of significant covariates and interactions, we now perform estimation and inference on them. From `summary(lm(...))` for `model_2b` (as illustrated in Appendix 2), we observe that the coefficients of several covariates cannot be properly estimated, nor can inference be drawn. This is not due to the lack of degrees of freedom, as discussed in the previous section. Instead, we experience exact collinearity between levels within categorical covariates – which suggests the use of Ridge to eliminate singularity and improve stability. In addition, the dimensionality of the model is still very large. It would be advisable to further reduce the number of covariates and improve model interpretability – which suggests the use of LASSO/sparse regression.

```
X = model.matrix(model_2b)
Y = as.matrix(auto["highway_mpg"])
```

Without better guidance on the choice of Ridge/LASSO and the penalty level, we run regressions with elastic net (which combines both Ridge & LASSO penalties) and perform grid search using `cv.glmnet` with α (the relative size of Ridge vs. LASSO) and λ (the penalty strength) as parameters.

```
set.seed(4)
alphas = seq(from = 0.0, to = 1.0, by = 0.02)
lbd_mins = c()
mse_mins = c()

par(mfrow = c(1, 2))
for (alpha in alphas) {
  cv_fit = cv.glmnet(X, Y, alpha = alpha)
  lbd_mins = c(lbd_mins, cv_fit$lambda.min)
  mse_mins = c(mse_mins, min(cv_fit$cvm))
  if (alpha == 0 || alpha == 1) {
    plot(cv_fit)
    if (alpha == 0) {
      title(paste0("Ridge"), line = 2.5)
    }
    if (alpha == 1) {
      title(paste0("LASSO"), line = 2.5)
    }
  }
}
```



We look for the α, λ combination that minimizes the out-of-sample mean-squared error. The selected $\hat{\alpha} = 0.80$ suggests a model that favors LASSO (sparsity) over Ridge (stability).

```
set.seed(10)
mse_min = which.min(mse_mins)
alp_min = format(round(alphas[which.min(mse_mins)], 2), nsmall = 2)
lbd_min = format(round(lbd_mins[which.min(mse_mins)], 2), nsmall = 2)
cat(paste0("Minimum MSE obtained with alpha=", alp_min),
    paste0("and lambda=", lbd_min))
```

```
## Minimum MSE obtained with alpha=0.80 and lambda=0.16
```

The coefficients estimated for this final model are summarized below, which indeed show a high degree of sparsity.

```
model_final = glmnet(X, Y, alpha=alp_min, lambda=lbd_min)
coef(model_final)
```

```
## 74 x 1 sparse Matrix of class "dgCMatrix"
##                                          s0
## (Intercept)                  50.7807489900
## (Intercept)                   .
## makeaudi                      .
## makebmw                       .
## makechevrolet                 3.2865653493
```

```
## makedodge                          .
## makehonda                          .
## makeisuzu                          .
## makejaguar                         0.5702753651
## makemazda                          .
## makemercedes-benz                  .
## makemercury                        .
## makemitsubishi                     .
## makenissan                         .
## makepeugot                         .
## makeplymouth                       .
## makeporsche                        .
## makerenault                        .
## makesaab                           .
## makesubaru                         .
## maketoyota                         0.0062401671
## makevolkswagen                     .
## makevolvo                          .
## fuel_typegas                       .
## wheel_base                         .
## length                            -0.0353622880
## width                              .
## height                             .
## curb_weight                       -0.0060050740
## num_of_cylindersfive              -1.3453858729
## num_of_cylindersfour               2.1582131610
## num_of_cylinderssix                .
## num_of_cylindersthree              8.7288883338
## num_of_cylinderstwelve            -0.9870988284
## num_of_cylinderstwo               -3.8194035254
## engine_typedohcv                   4.6660105543
## engine_typel                       .
## engine_typeohc                     0.9642509912
## engine_typeohcf                    .
## engine_typeohcv                   -0.1144475457
## engine_typerotor                  -0.8364899671
## fuel_system2bbl                    1.6293261272
## fuel_system4bbl                   -0.0957731488
## fuel_systemidi                     .
## fuel_systemmfi                     .
## fuel_systemmpfi                    .
## fuel_systemspdi                    .
## fuel_systemspfi                    .
## aspirationturbo                   -1.7704942376
## bore                              -2.3568724706
## compression_rate                   0.6437671620
## makeaudi:compression_rate         -0.0300234964
## makebmw:compression_rate           .
## makechevrolet:compression_rate     0.1474550149
## makedodge:compression_rate         .
## makehonda:compression_rate         .
## makeisuzu:compression_rate         0.0918918060
## makejaguar:compression_rate        .
## makemazda:compression_rate         .
```

```
## makemercedes-benz:compression_rate  .
## makemercury:compression_rate        .
## makemitsubishi:compression_rate     .
## makenissan:compression_rate         0.0970064564
## makepeugot:compression_rate         .
## makeplymouth:compression_rate       0.0008532653
## makeporsche:compression_rate        .
## makerenault:compression_rate        .
## makesaab:compression_rate           .
## makesubaru:compression_rate         .
## maketoyota:compression_rate         .
## makevolkswagen:compression_rate     0.0344459517
## makevolvo:compression_rate          .
## length:compression_rate             .
## fuel_typegas:compression_rate       .
```

5.2 Pairwise Comparisions

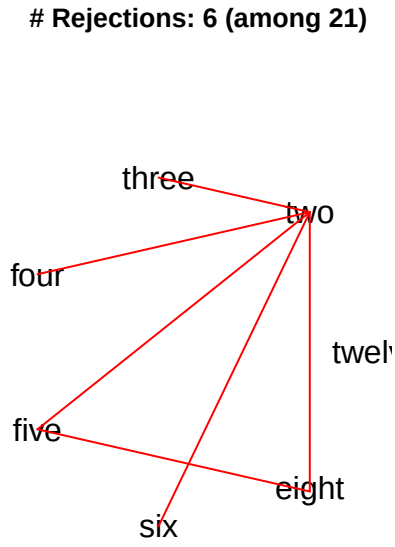
In our final model, we have found a set of significant categorical covariates `make`, `num_of_cylinders` and `fuel_system` (suggesting that the levels within each covariate do not all share the same mean). We are also interested in knowing for which pairs of levels are the effects different for each categorical covariate, which we now investigate with Tukey's HSD test at the customary 0.05 significance level.

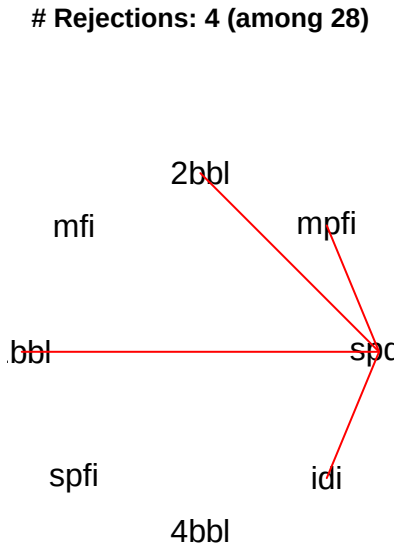
```
plot_tukey = function(index, p_adj) {
  K = length(index)
  plot(
    -1:1,
    -1:1,
    type = "n",
    axes = FALSE,
    xlab = "",
    ylab = "",
    asp = 1
  )
  title(
    main = paste0(
      "# Rejections: ",
      sum(p_adj <= 0.05),
      " (",
      "among ",
      choose(length(index), 2),
      ")"
    ),
    cex.main = 0.8
  )

  # plot the texts
  for (i in 1:K) {
    theta = i / K * 2 * pi
    text(cos(theta), sin(theta), index[i])
  }

  # plot the segments
```

```
  for (i in 1:K) {
    for (j in 1:K) {
      theta_i = i / K * 2 * pi
      theta_j = j / K * 2 * pi
      str = paste0(index[i], "-", index[j])
      if (str %in% names(p_adj) & p_adj[str] <= 0.05) {
        segments(cos(theta_i),
                 sin(theta_i),
                 cos(theta_j),
                 sin(theta_j),
                 col = "red")
      }
    }
  }
}
```

In the plots below, we draw a line between each pair of levels for which the Tukey's HSD test is rejected. We observed that for the `num_of_cylinders`, the main difference comes from level `num_of_cylinders=two` versus the others. For `fuel_system`, the difference mainly comes from `fuel_system=spdi` against the rest.

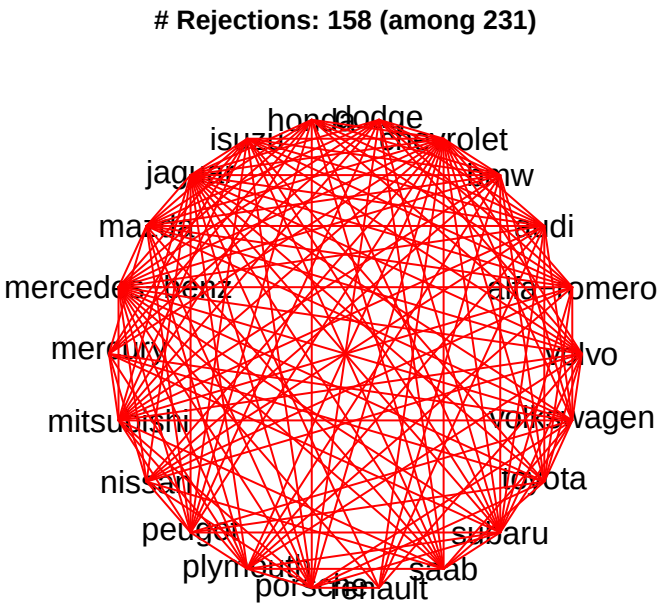
```
par(mfrow=c(1, 2))

# num_of_cylinders
index = c("two", "three", "four", "five", "six", "eight", "twelve")
tukey = TukeyHSD(aov(model_2b), "num_of_cylinders")
p_adj = tukey[[names(tukey)]][, "p adj"]
plot_tukey(index, p_adj)

# fuel_system
index = unique(auto[, "fuel_system"])
tukey = TukeyHSD(aov(model_2b), "fuel_system")
p_adj = tukey[[names(tukey)]][, "p adj"]
plot_tukey(index, p_adj)
```





For the `make` variable, which has 22 distinct levels, a very large fractions of pairwise comparisons are significant. This agrees with our previous analysis that the covariate is highly significant with great explanatory power.

```
# make
index = unique(auto[, "make"])
tukey = TukeyHSD(aov(model_2b), "make")
p_adj = tukey[[names(tukey)]][, "p adj"]
plot_tukey(index, p_adj)
```



6. Conclusion

In this report, we have used the `auto` data set to build a linear model for the response `highway_mpg` – the car's performance in miles per gallon during highway driving, with all the other variables as potential covariates. We have performed diagnostics to find outliers with high studentized residuals or Cook's distance. We then carried out model selection to find a set of significant covariates `make`, `fuel_type`, `wheel_base`, `length`, `width`, `height`, `curb_weight`, `num_of_cylinders`, `engine_type`, `fuel_system`, `aspiration`, `bore`, `compression_rate` and interactions `make:compression_rate`, `length:compression_rate`, `fuel_type:compression_rate`. In the end, we used cross-validation to obtain an elastic net model with the selected covariates that have great out-of-sample predictive power.

Further Work

Selective Inference: One drawback of our methodology is that we have used the same data set for model selection and estimation/inference of model coefficients (both of which are run on the whole data set). The significance of some covariates (interaction terms, in particular) is very near to the 0.05 significance level, suggesting only weak associations with the response, which may in fact be false positive. A better approach would be to split the data set into a training set and a validation set (e.g. 70/30 split), using the first group for selection and the subsequent group for estimation/inference. However, given the small data set we have (with only 203 observations), such an approach may introduce undesirably high variance in the estimated coefficients. Cross-validation will alleviate this issue.

Collinearity: Early in our exploratory data analysis, we have seen many pairs of collinear covariates in our data, whose effects on the response tends to be quite similar. However, we notice that in our model selection

procedure, especially when adding interaction terms, the terms being included are highly dependent upon the order in which they appear in the F-test (i.e. if a covariate is already added to the model, covariates with similar effects added later are likely to be insignificant). We could run our model selection with different ordering of the covariates, and ideally come up with a model with the best explanatory power and physical interpretation.

Inference for elastic net: Another drawback of our investigation is that our final model is fitted with elastic net which does not come directly with estimates for standard error / confidence interval. Further work can be done to use bootstrap to create inference for the estimated coefficients.

Appendix 1: Coefficients of the Preliminary Model

```
summary(model_out)
```

```
##
## Call:
## lm(formula = highway_mpg ~ ., data = auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.8003 -0.9383 -0.0012  0.9721  9.8294
##
## Coefficients: (3 not defined because of singularities)
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)              3.811e+01  2.393e+01   1.593  0.11335
## makeaudi                 1.226e+00  3.107e+00   0.395  0.69376
## makebmw                  5.265e+00  3.197e+00   1.647  0.10175
## makechevrolet            7.418e+00  2.930e+00   2.532  0.01242 *
## makedodge                3.118e+00  2.525e+00   1.235  0.21890
## makehonda               -9.700e-01  3.001e+00  -0.323  0.74696
## makeisuzu                4.804e+00  2.672e+00   1.798  0.07429 .
## makejaguar               7.221e+00  3.821e+00   1.890  0.06078 .
## makemazda                2.175e+00  2.327e+00   0.934  0.35167
## makemercedes-benz        2.657e+00  3.452e+00   0.770  0.44265
## makemercury              2.399e+00  4.012e+00   0.598  0.55082
## makemitsubishi           2.789e+00  2.507e+00   1.112  0.26779
## makenissan               3.615e+00  2.268e+00   1.594  0.11321
## makepeugot               4.234e+00  2.846e+00   1.487  0.13907
## makeplymouth             3.827e+00  2.481e+00   1.542  0.12519
## makeporsche             -1.332e+00  4.060e+00  -0.328  0.74326
## makerenault              1.725e+00  2.990e+00   0.577  0.56496
## makesaab                 1.634e+00  2.792e+00   0.585  0.55931
## makesubaru              -4.142e+00  4.662e+00  -0.889  0.37574
## maketoyota               3.122e+00  2.116e+00   1.475  0.14243
## makevolkswagen           2.536e+00  2.410e+00   1.052  0.29436
## makevolvo                5.112e+00  2.944e+00   1.736  0.08467 .
## fuel_typegas             1.833e+01  8.989e+00   2.040  0.04323 *
## wheel_base              -1.679e-01  1.228e-01  -1.368  0.17341
## length                  -1.233e-01  6.816e-02  -1.809  0.07253 .
## width                    6.136e-01  3.025e-01   2.028  0.04436 *
## height                  -1.879e-01  1.999e-01  -0.940  0.34880
## curb_weight             -5.819e-03  2.287e-03  -2.544  0.01200 *
## num_of_doorstwo          1.048e-01  6.542e-01   0.160  0.87294
## body_stylehardtop        1.106e+00  1.710e+00   0.647  0.51892
## body_stylehatchback      2.144e+00  1.601e+00   1.339  0.18258
## body_stylesedan          3.055e+00  1.723e+00   1.773  0.07826 .
## body_stylewagon          3.562e+00  1.806e+00   1.972  0.05049 .
## drive_wheelsfwd          1.085e+00  1.226e+00   0.885  0.37776
## drive_wheelsrwd         -4.378e-01  1.607e+00  -0.272  0.78570
## num_of_cylindersfive     7.968e-01  3.968e+00   0.201  0.84114
## num_of_cylindersfour     3.737e+00  4.874e+00   0.767  0.44456
## num_of_cylinderssix     -1.127e+00  3.687e+00  -0.306  0.76029
## num_of_cylindersthree    1.016e+01  5.727e+00   1.774  0.07822 .
```

```
## num_of_cylinderstwelve -1.502e+01  7.013e+00  -2.142  0.03386 *
## num_of_cylinderstwo    -4.382e+00  6.098e+00  -0.719  0.47351
## engine_typedohcv        2.790e+00  6.550e+00   0.426  0.67082
## engine_typel                   NA         NA      NA       NA
## engine_typeohc         -4.166e-01  1.665e+00  -0.250  0.80282
## engine_typeohcf         4.040e+00  3.695e+00   1.093  0.27610
## engine_typeohcv        -9.153e-01  1.706e+00  -0.537  0.59238
## engine_typerotor               NA         NA      NA       NA
## fuel_system2bbl        -4.279e+00  2.025e+00  -2.113  0.03637 *
## fuel_system4bbl        -4.946e+00  3.520e+00  -1.405  0.16217
## fuel_systemidi                 NA         NA      NA       NA
## fuel_systemmfi         -7.534e+00  3.617e+00  -2.083  0.03902 *
## fuel_systemmpfi        -5.258e+00  2.138e+00  -2.459  0.01513 *
## fuel_systemspdi        -5.925e+00  2.506e+00  -2.364  0.01939 *
## fuel_systemspfi        -7.644e+00  3.657e+00  -2.090  0.03835 *
## aspirationturbo        -2.652e+00  1.096e+00  -2.420  0.01675 *
## engine_size            -1.223e-03  3.422e-02  -0.036  0.97153
## bore                   -2.751e+00  2.559e+00  -1.075  0.28429
## stroke                 -2.496e-01  1.377e+00  -0.181  0.85646
## compression_rate        1.602e+00  6.605e-01   2.425  0.01655 *
## peak_rpm               -2.726e-03  8.751e-04  -3.115  0.00222 **
## horsepower              2.758e-02  3.319e-02   0.831  0.40732
## normalized_losses       1.326e-04  1.121e-02   0.012  0.99058
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.427 on 144 degrees of freedom
## Multiple R-squared:  0.9111, Adjusted R-squared:  0.8753
## F-statistic: 25.45 on 58 and 144 DF,  p-value: < 2.2e-16
```

Appendix 2: Coefficients of the Selected Model with Interactions

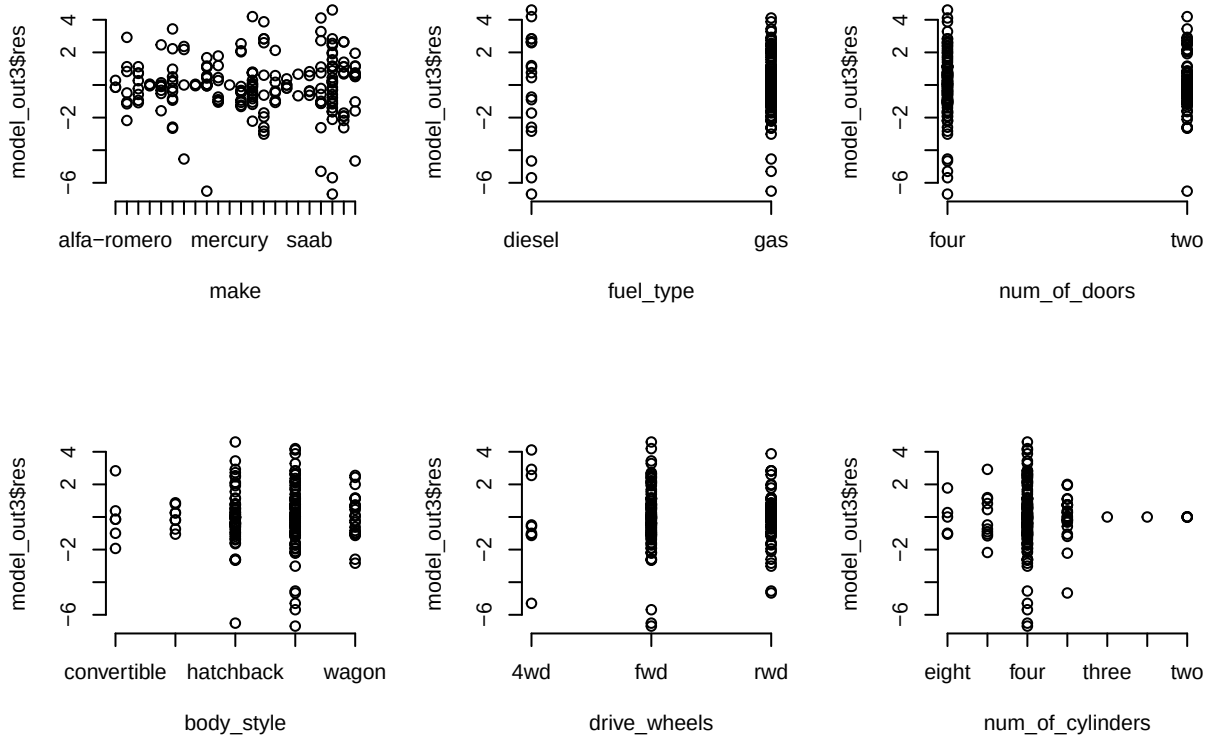
```
summary(model_2b)
```

```
##
## Call:
## lm(formula = formula_step, data = auto[, c("highway_mpg", selected)])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.3281 -0.4832 -0.0433  0.6303  6.0650
##
## Coefficients: (9 not defined because of singularities)
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       6.720e+02  2.210e+02   3.040 0.002837 **
## makeaudi                          2.013e+01  9.632e+00   2.090 0.038482 *
## makebmw                          -2.449e+00  1.377e+01  -0.178 0.859054
## makechevrolet                     5.123e+00  1.785e+00   2.869 0.004770 **
## makedodge                         1.303e+00  1.061e+01   0.123 0.902438
## makehonda                        -2.730e+01  1.679e+01  -1.626 0.106192
## makeisuzu                        -6.119e+01  1.875e+01  -3.264 0.001392 **
## makejaguar                        4.636e+00  2.216e+00   2.093 0.038247 *
## makemazda                         1.703e+01  6.330e+00   2.691 0.008019 **
## makemercedes-benz                 1.699e+01  1.283e+02   0.132 0.894856
## makemercury                       1.462e+00  2.462e+00   0.594 0.553699
## makemitsubishi                   -4.554e+00  1.331e+01  -0.342 0.732698
## makenissan                        2.083e+01  7.312e+00   2.849 0.005068 **
## makepeugot                        3.499e+01  1.144e+01   3.059 0.002675 **
## makeplymouth                     -7.044e+00  1.336e+01  -0.527 0.598791
## makeporsche                      -1.427e+00  2.371e+00  -0.602 0.548118
## makerenault                       7.032e-01  1.780e+00   0.395 0.693381
## makesaab                          1.819e+02  5.719e+01   3.180 0.001822 **
## makesubaru                        7.350e+00  9.961e+00   0.738 0.461871
## maketoyota                        1.209e+01  3.539e+00   3.418 0.000834 ***
## makevolkswagen                   -4.049e+00  2.452e+00  -1.651 0.100969
## makevolvo                         9.670e-01  1.668e+00   0.580 0.563082
## fuel_typegas                     -6.755e+02  2.045e+02  -3.303 0.001222 **
## wheel_base                       -6.286e-02  8.918e-02  -0.705 0.482069
## length                            2.873e-01  7.666e-02   3.748 0.000263 ***
## width                             1.389e-01  2.159e-01   0.644 0.520857
## height                           -1.151e-01  1.179e-01  -0.976 0.330950
## curb_weight                      -5.732e-03  1.309e-03  -4.378 2.37e-05 ***
## num_of_cylindersfive             -6.420e+00  7.954e+01  -0.081 0.935782
## num_of_cylindersfour             -1.578e+00  7.959e+01  -0.020 0.984213
## num_of_cylinderssix              -2.679e+00  7.945e+01  -0.034 0.973152
## num_of_cylindersthree             1.095e+01  7.965e+01   0.137 0.890888
## num_of_cylinderstwelve           -1.248e+01  7.926e+01  -0.157 0.875133
## num_of_cylinderstwo              -5.952e+00  7.942e+01  -0.075 0.940376
## engine_typedohcv                  5.981e+00  7.950e+01   0.075 0.940138
## engine_typel                             NA         NA      NA       NA
## engine_typeohc                    4.608e+00  9.800e-01   4.702 6.24e-06 ***
## engine_typeohcf                   2.999e+00  2.286e+00   1.312 0.191646
## engine_typeohcv                   5.091e-01  1.114e+00   0.457 0.648465
```

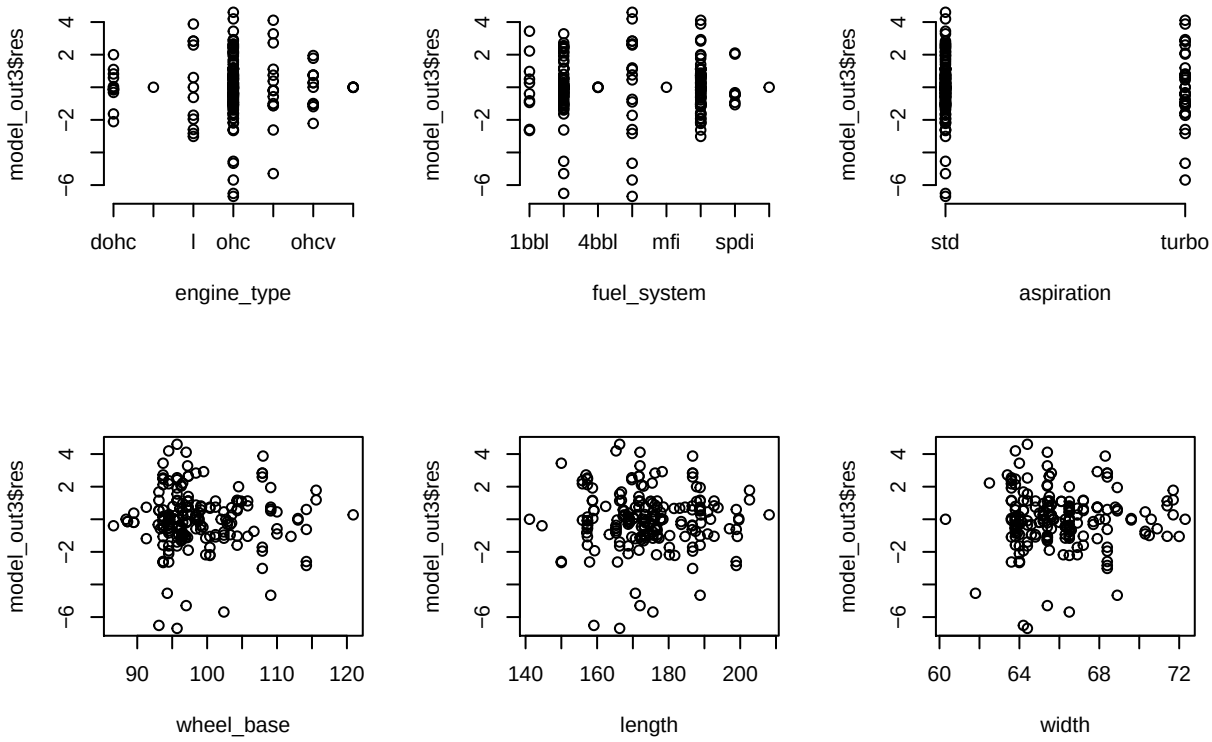
```
## engine_typerotor                          NA         NA      NA       NA
## fuel_system2bbl                     -2.251e+00  1.325e+00  -1.699 0.091678 .
## fuel_system4bbl                     -3.596e+00  2.328e+00  -1.545 0.124646
## fuel_systemidi                            NA         NA      NA       NA
## fuel_systemmfi                      -2.383e+00  3.108e+00  -0.767 0.444510
## fuel_systemmpfi                     -3.049e+00  1.396e+00  -2.184 0.030640 *
## fuel_systemspdi                     -1.052e+00  2.579e+00  -0.408 0.684072
## fuel_systemspfi                     -6.387e+00  2.393e+00  -2.669 0.008542 **
## aspirationturbo                     -2.149e+00  7.248e-01  -2.965 0.003578 **
## bore                                -4.866e+00  1.784e+00  -2.727 0.007228 **
## compression_rate                    -2.355e+01  8.843e+00  -2.663 0.008675 **
## makeaudi:compression_rate           -2.466e+00  1.049e+00  -2.350 0.020201 *
## makebmw:compression_rate             1.579e-01  1.549e+00   0.102 0.918968
## makechevrolet:compression_rate              NA         NA      NA       NA
## makedodge:compression_rate          -3.939e-02  1.171e+00  -0.034 0.973219
## makehonda:compression_rate           2.595e+00  1.851e+00   1.401 0.163346
## makeisuzu:compression_rate           6.910e+00  2.031e+00   3.402 0.000879 ***
## makejaguar:compression_rate                 NA         NA      NA       NA
## makemazda:compression_rate          -1.892e+00  6.771e-01  -2.794 0.005953 **
## makemercedes-benz:compression_rate -2.381e+00  6.007e+00  -0.396 0.692473
## makemercury:compression_rate                NA         NA      NA       NA
## makemitsubishi:compression_rate      5.972e-01  1.474e+00   0.405 0.686064
## makenissan:compression_rate         -2.143e+00  7.741e-01  -2.768 0.006432 **
## makepeugot:compression_rate         -3.823e+00  1.343e+00  -2.847 0.005100 **
## makeplymouth:compression_rate        9.001e-01  1.459e+00   0.617 0.538324
## makeporsche:compression_rate                NA         NA      NA       NA
## makerenault:compression_rate                NA         NA      NA       NA
## makesaab:compression_rate           -1.973e+01  6.228e+00  -3.167 0.001901 **
## makesubaru:compression_rate         -6.036e-01  1.100e+00  -0.549 0.584234
## maketoyota:compression_rate         -1.214e+00  3.575e-01  -3.396 0.000897 ***
## makevolkswagen:compression_rate      3.566e-01  1.900e-01   1.877 0.062630 .
## makevolvo:compression_rate                  NA         NA      NA       NA
## length:compression_rate             -2.799e-02  6.736e-03  -4.155 5.72e-05 ***
## fuel_typegas:compression_rate        3.046e+01  9.104e+00   3.346 0.001060 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.615 on 136 degrees of freedom
## Multiple R-squared:  0.9594, Adjusted R-squared:  0.9405
## F-statistic: 50.96 on 63 and 136 DF,  p-value: < 2.2e-16
```

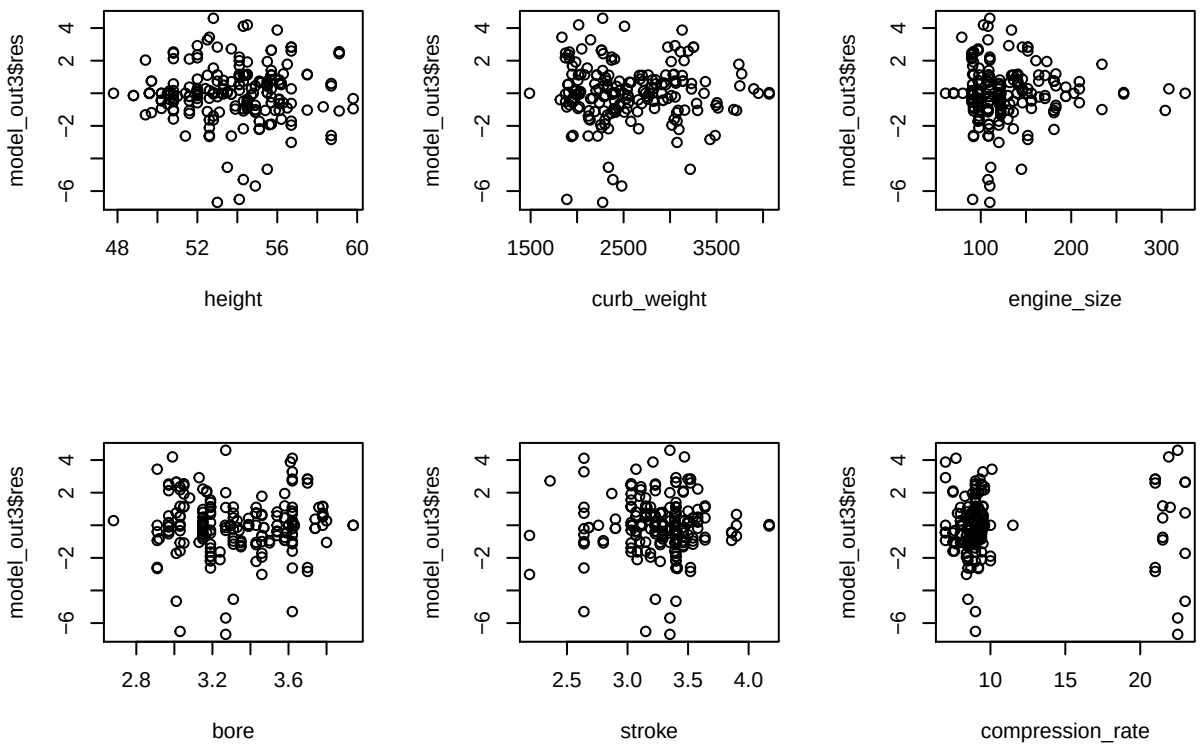
Appendix 3: Residual Plots for Diagonistics

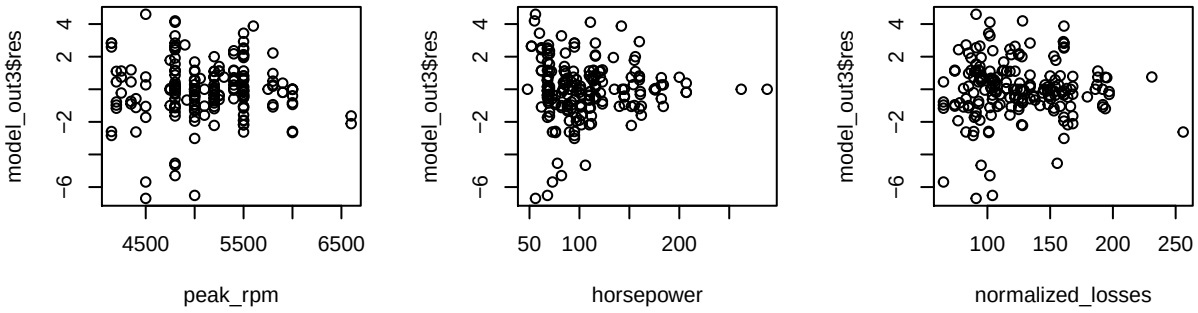
```
par(mfrow = c(2, 3))
for (i in factor_col) {
  plot(as.numeric(auto[, i]),
       model_out3$res,
       axes = FALSE,
       xlab = colnames(auto)[i])
  axis(
    side = 1,
    at = 1:length(levels(auto[, i])),
    lab = levels(auto[, i])
  )
  axis(side = 2)
}
```




```
for (i in num_col) {
  plot(auto[, i], model_out3$res, xlab = colnames(auto)[i])
}
```

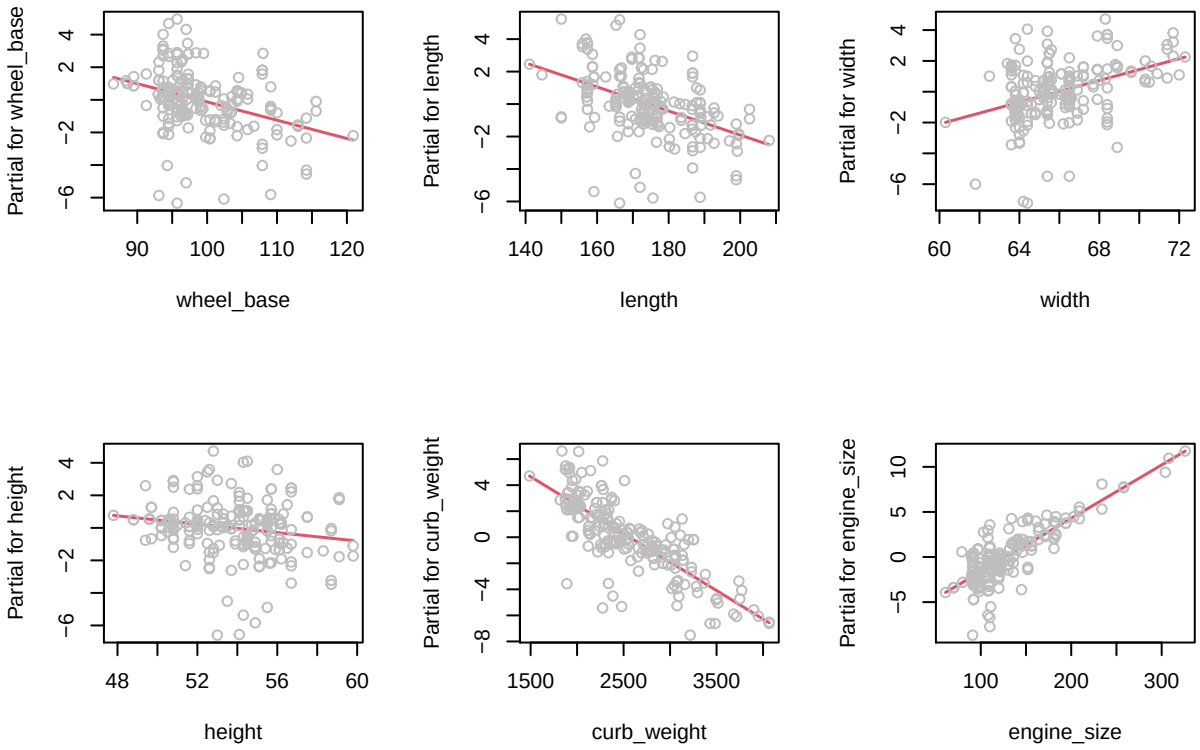

model_out3$res
−6
−2
2
4
dohc
l
ohc
ohcv
engine_type
1bbl
4bbl
mfi
spdi
fuel_system
std
turbo
aspiration
90
100
110
120
wheel_base
140
160
180
200
length
60
64
68
72
width

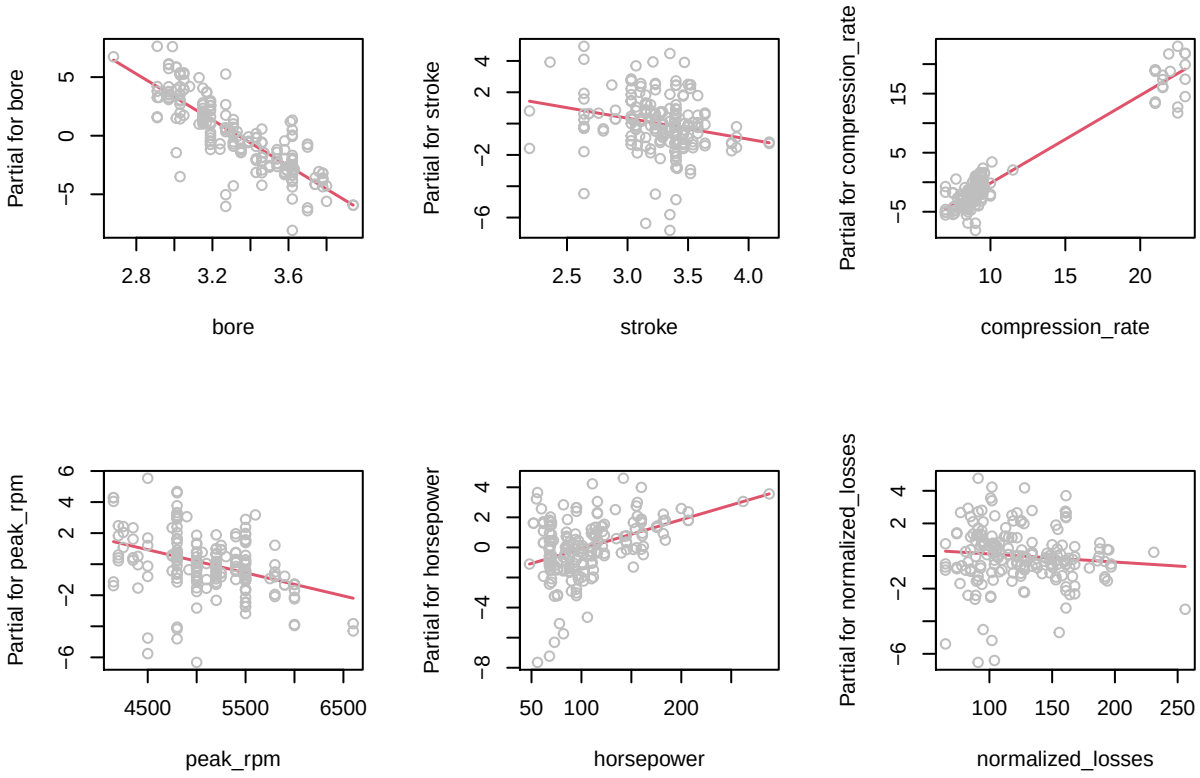

model_out3$res
height
curb_weight
engine_size
bore
stroke
compression_rate


model_out3$res
4
2
−2
−6
4500
5500
6500
peak_rpm
model_out3$res
4
2
−2
−6
50
100
200
horsepower
model_out3$res
4
2
−2
−6
100
150
200
250
normalized_losses

Appendix 4: Partial Residual Plots for Diagonistics

```
par(mfrow = c(2, 3))
for (i in num_col) {
  termplot(model_out3, partial.resid = TRUE, terms = i)
}
```




Partial for bore
bore
Partial for stroke
stroke
Partial for compression_rate
compression_rate
Partial for peak_rpm
peak_rpm
Partial for horsepower
horsepower
Partial for normalized_losses
normalized_losses