Appendix: Pricing and Market Analysis for Used BMW Cars

Lab2 Group A: Lee, Joshua; Liu, Kaiyi; Pulsone, Nathaniel; Wang, Mengyao; Xu, Zexian; Yang, Xiaojing

Contents

Load in packages

```
mylibrary <- c("tidyverse", "cowplot", "GGally", "MASS", "ggplot2",
    "glmnet", "data.table", "car", "MLmetrics", "patchwork")
invisible(lapply(mylibrary, library, character.only = TRUE))
```

Function for drawing diagnostic plots

```
diagPlot <- function(model) {
    p_resid <- ggplot(model, aes(.fitted, .resid)) + geom_point() +
        stat_smooth(method = "loess") + geom_hline(yintercept = 0,
        col = "red", linetype = "dashed") + xlab("Fitted values") +
        ylab("Residuals") + ggtitle("Residual vs Fitted Plot") +
        theme_bw()

    p_QQ <- ggplot(model, aes(sample = .stdresid)) + stat_qq() +
        stat_qq_line() + xlab("Theoretical Quantiles") + ylab("Standardized Residuals")
         ↪  +
        ggtitle("Normal Q-Q") + theme_bw()
```

```
    p_SL <- ggplot(model, aes(.fitted, sqrt(abs(.stdresid)))) +
        geom_point(na.rm = TRUE) + stat_smooth(method = "loess",
        na.rm = TRUE) + xlab("Fitted Value") + ylab(expression(sqrt("|Standardized
         ↪  residuals|"))) +
        ggtitle("Scale-Location") + theme_bw()

    p_lev <- ggplot(model, aes(.hat, .stdresid)) + geom_point(na.rm = TRUE) +
        stat_smooth(method = "loess", na.rm = TRUE) + xlab("Leverage") +
        ylab("Standardized Residuals") + ggtitle("Residual vs Leverage Plot") +
        scale_size_continuous("Cook's Distance", range = c(1,
            5)) + theme_bw() + theme(legend.position = "bottom")

    ## combine plots
    plot_grid(p_resid, p_QQ, p_SL, p_lev, align = "h")
}
```

read in dataset

```
dat_bmw <- read.csv("BMWpricing_updated.csv")
```

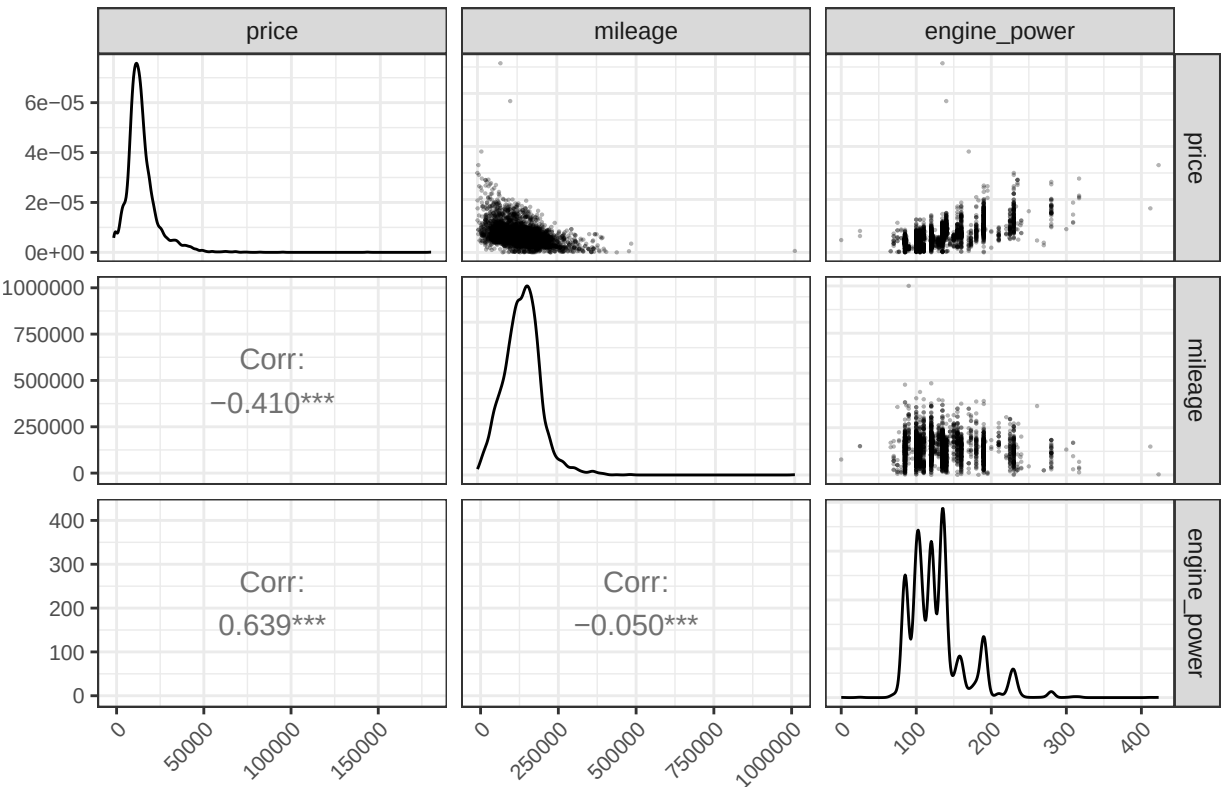
Data

Data Overview

We first drew the scatterplot matrix between price, mileage, and engine power.

```
### Create a data frame only has the variable of interests
dat_bmw2 <- dat_bmw[, c("price", "mileage", "engine_power")]
### Drop the data point with NA
dat_bmw2 <- dat_bmw2[complete.cases(dat_bmw2), ]

ggpairs(dat_bmw2, upper = list(continuous = wrap("points", alpha = 0.3,
    size = 0.1)), lower = list(continuous = wrap("cor", size = 4))) +
    labs(title = "Distribution and Correlation between Price, Mileage, and Engine
     ↪  Power") +
    theme_bw() + theme(axis.text.x = element_text(angle = 45,
    hjust = 1, size = 8), axis.text.y = element_text(size = 8),
    title = element_text(size = 12))
```



Then we investigated the distribution of price, the relationship between price and registribution date, and the sales by action date.

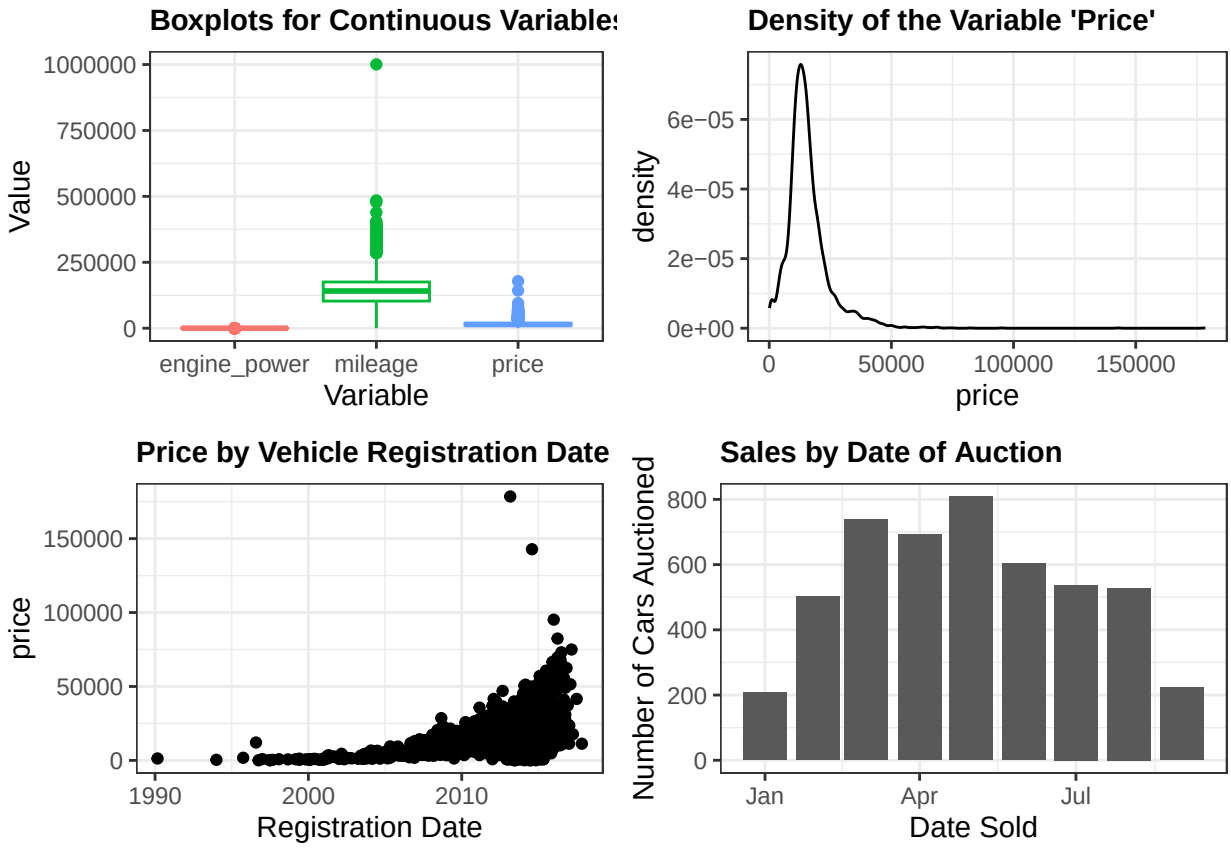
```
p_box_conti <- dat_bmw[, c("price", "mileage", "engine_power")] |>
    pivot_longer(everything(), values_to = "Value", names_to = "Variable") |>
    ggplot() + geom_boxplot(aes(x = Variable, y = Value, color = Variable)) +
    labs(title = "Boxplots for Continuous Variables") + theme_bw() +
    theme(plot.title = element_text(size = 11, face = "bold"),
        legend.position = "none")

p_price <- dat_bmw |>
    ggplot() + geom_density(aes(x = price)) + labs(title = "Density of the Variable
     ↪  'Price'") +
    theme_bw() + theme(plot.title = element_text(size = 11, face = "bold"))

p_price_date <- dat_bmw |>
    ggplot(aes(x = as.Date(registration_date, format = "%m/%d/%Y"),
        y = price)) + geom_point() + labs(title = "Price by Vehicle Registration Date")
         ↪  +
    xlab("Registration Date") + theme_bw() + theme(plot.title = element_text(size = 11,
    face = "bold"))

p_sale_Date <- dat_bmw |>
    ggplot(aes(x = as.Date(sold_at, format = "%m/%d/%Y"))) +
    geom_bar() + labs(title = "Sales by Date of Auction") + xlab("Date Sold") +
    ylab("Number of Cars Auctioned") + theme_bw() + theme(plot.title =
     ↪  element_text(size = 11,
    face = "bold"))
```

```
plot_grid(p_box_conti, p_price, p_price_date, p_sale_Date, ncol = 2)
```

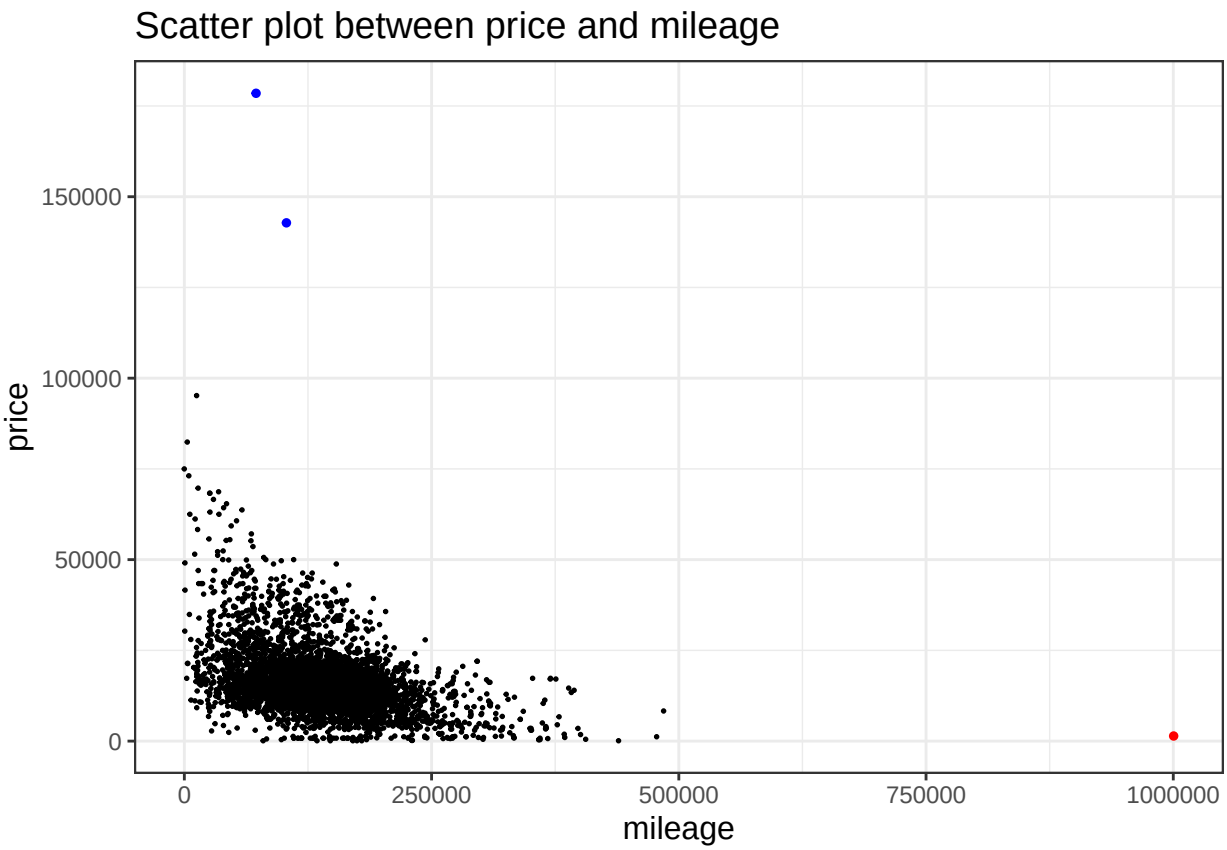


Missing data and implausible value handling

We drew the scatter plot between price and mileage to explore potential outliers. We highlighted these potential outliers.

```
scatterPlotPriceMileage <- ggplot(data = dat_bmw2, mapping = aes(x = mileage, y =
 ↪  price)) +
  geom_point(pch=19, cex=0.3) +  # Default points
  geom_point(data = subset(dat_bmw2, mileage > 500000), mapping = aes(x = mileage, y =
   ↪  price), pch=19, cex=1, color = "red") +  # Highlight points with mileage >
   ↪  500000 in red
  geom_point(data = subset(dat_bmw2, price > 100000), mapping = aes(x = mileage, y =
   ↪  price), pch=19, cex=1, color = "blue") + # Highlight points with price > 100000
   ↪  in blue
  labs(title = "Scatter plot between price and mileage") +
  theme_bw() + theme(title = element_text(size=12))

scatterPlotPriceMileage
```



Before, we fit the model, we removed the outliers we observed from the scatter plot (price > 100000 and mileage > 500000). We reasonably conclude that these points are errors in data entry that skew the model too heavily.

```
dat_bmw_clean <- dat_bmw |>
    filter(mileage < 5e+05 & mileage > 0) |>
    filter(price < 1e+05) |>
    filter(engine_power != 0)

head(dat_bmw_clean)
```

```
##   maker_key model_key mileage engine_power registration_date   fuel paint_color
## 1       BMW       118  140411          100          2/1/2012 diesel       black
## 2       BMW        M4   13929          317          4/1/2016 petrol        grey
## 3       BMW       320  183297          120          4/1/2012 diesel       white
## 4       BMW       420  128035          135          7/1/2014 diesel         red
## 5       BMW       425   97097          160         12/1/2014 diesel      silver
## 6       BMW       335  152352          225          5/1/2011 petrol       black
##      car_type feature_1 feature_2 feature_3 feature_4 feature_5 feature_6
## 1 convertible      TRUE      TRUE     FALSE     FALSE      TRUE      TRUE
## 2 convertible      TRUE      TRUE     FALSE     FALSE     FALSE      TRUE
## 3 convertible     FALSE     FALSE     FALSE     FALSE      TRUE     FALSE
## 4 convertible      TRUE      TRUE     FALSE     FALSE      TRUE      TRUE
## 5 convertible      TRUE      TRUE     FALSE     FALSE     FALSE      TRUE
## 6 convertible      TRUE      TRUE     FALSE     FALSE      TRUE      TRUE
##   feature_7 feature_8 price  sold_at obs_type
## 1      TRUE     FALSE 11300 1/1/2018 Training
## 2      TRUE      TRUE 69700 2/1/2018 Training
## 3      TRUE     FALSE 10200 2/1/2018 Training
```

```
## 4      TRUE      TRUE 25100 2/1/2018 Training
## 5      TRUE      TRUE 33400 4/1/2018 Training
## 6      TRUE      TRUE 17100 2/1/2018 Training
```

We regrouped the variable "model_key" based on cars' series.

```
### Create Model series variable
dat_bmw_clean <- dat_bmw_clean %>%
    mutate(model_series = case_when(grepl("^1", model_key) ~
        "1_Series", grepl("^2", model_key) ~ "2_Series", grepl("^3",
        model_key) ~ "3_Series", grepl("^4", model_key) ~ "4_Series",
        grepl("^5", model_key) ~ "5_Series", grepl("^7", model_key) ~
            "7_Series", grepl("^M|M$", model_key) ~ "M_Power",
        model_key %in% c("X1") ~ "X1", model_key %in% c("X3") ~
            "X3", model_key %in% c("X5") ~ "X5", model_key %in%
            c("X6") ~ "X6", TRUE ~ "Other"))
```

```
dat_bmw_clean$sold_at <- as.Date(dat_bmw_clean$sold_at, format = "%m/%d/%Y")
dat_bmw_clean$registration_date <- as.Date(dat_bmw_clean$registration_date,
    format = "%m/%d/%Y")
dat_bmw_clean$age <- as.numeric((dat_bmw_clean$sold_at -
 ↪  dat_bmw_clean$registration_date)/365.25)
```

Modelling and Analysis for the Full Data Set

Question 1

To explore which factor between mileage and engine power impacts the price of the car the most, we fit the baseline model: $price = \beta_0 + \beta_1 mileage + \beta_2 engine\ power + \epsilon$.

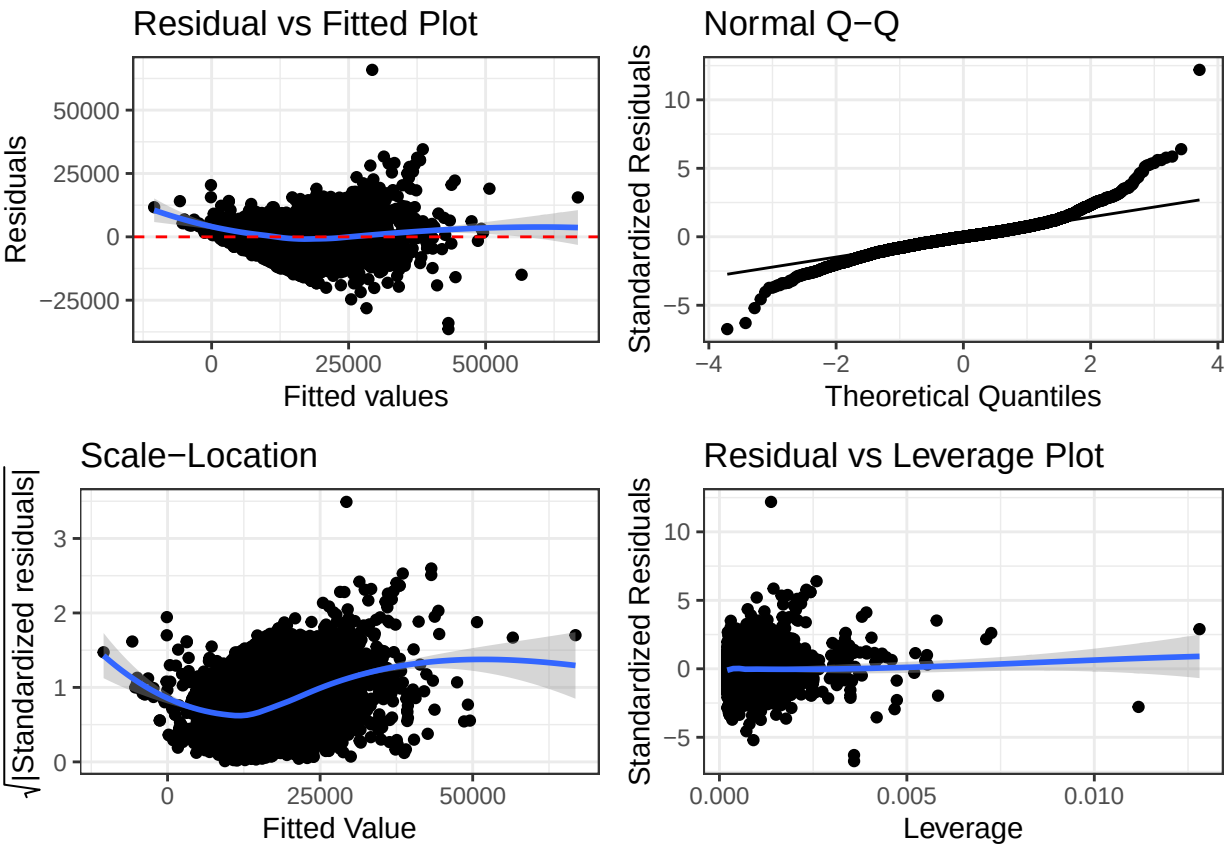
```
baseline_model <- lm(price ~ mileage + engine_power, data = dat_bmw_clean)
summary(baseline_model)
```

```
##
## Call:
## lm(formula = price ~ mileage + engine_power, data = dat_bmw_clean)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -36415  -2785    -73   2546  65891
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.193e+03  3.345e+02   15.53   <2e-16 ***
## mileage      -5.883e-02  1.322e-03  -44.49   <2e-16 ***
## engine_power  1.462e+02  2.000e+00   73.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5410 on 4835 degrees of freedom
```

```
## Multiple R-squared:  0.6127, Adjusted R-squared:  0.6125
## F-statistic:  3824 on 2 and 4835 DF,  p-value: < 2.2e-16
```

```
diagPlot(baseline_model)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



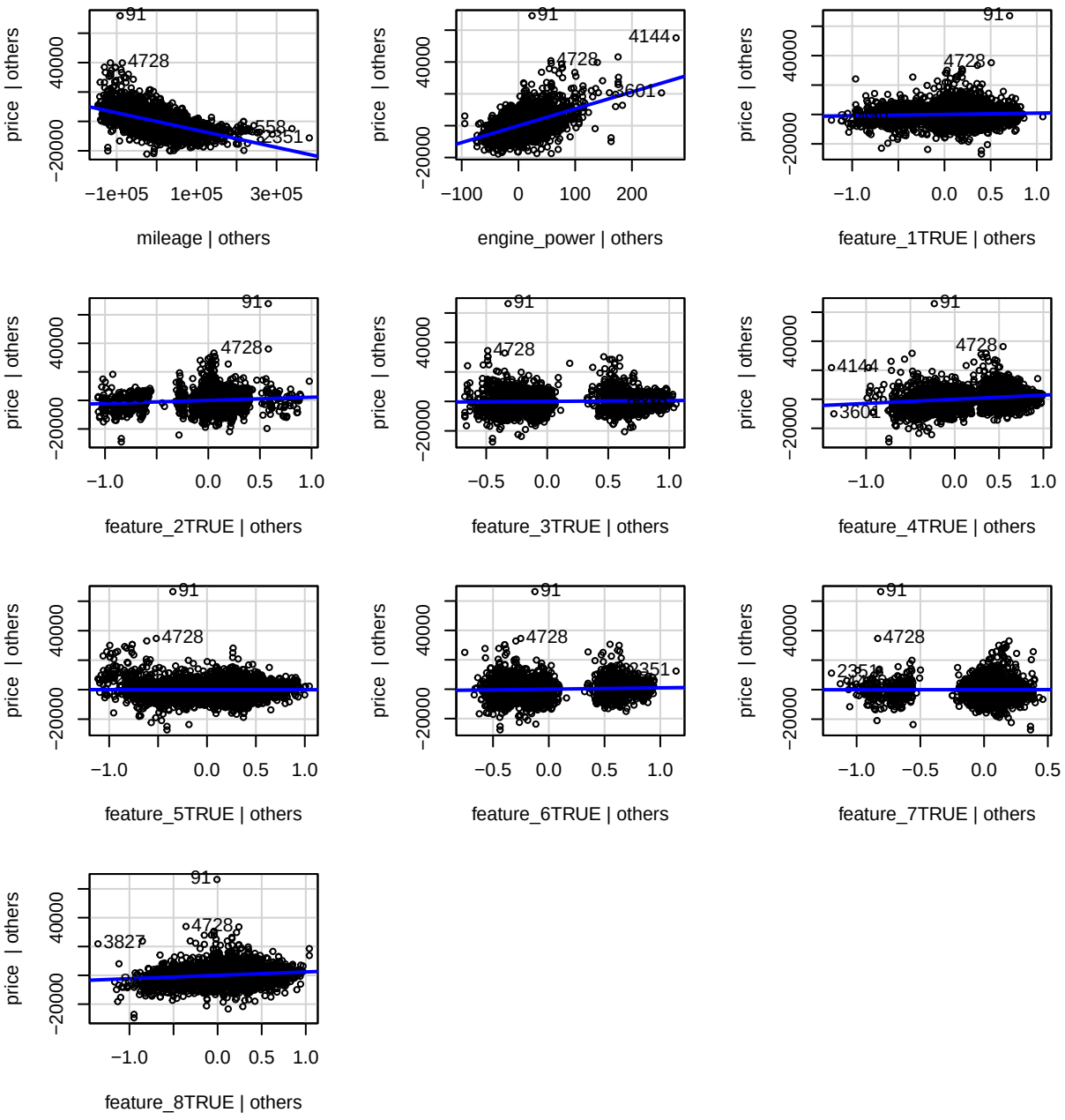
Question 2

We utilized added variable plots to solve our second question: "Which categorical predictors explain a significant amount of variance in auction price, after controlling for mileage, engine power?".

We first investigated if any feature variables could explain a significant amount of variance in auction price.

```
feature_model <- lm(price ~ mileage + engine_power + feature_1 +
    feature_2 + feature_3 + feature_4 + feature_5 + feature_6 +
    feature_7 + feature_8, data = dat_bmw_clean)

avPlots(feature_model, layout = c(4, 3))
```



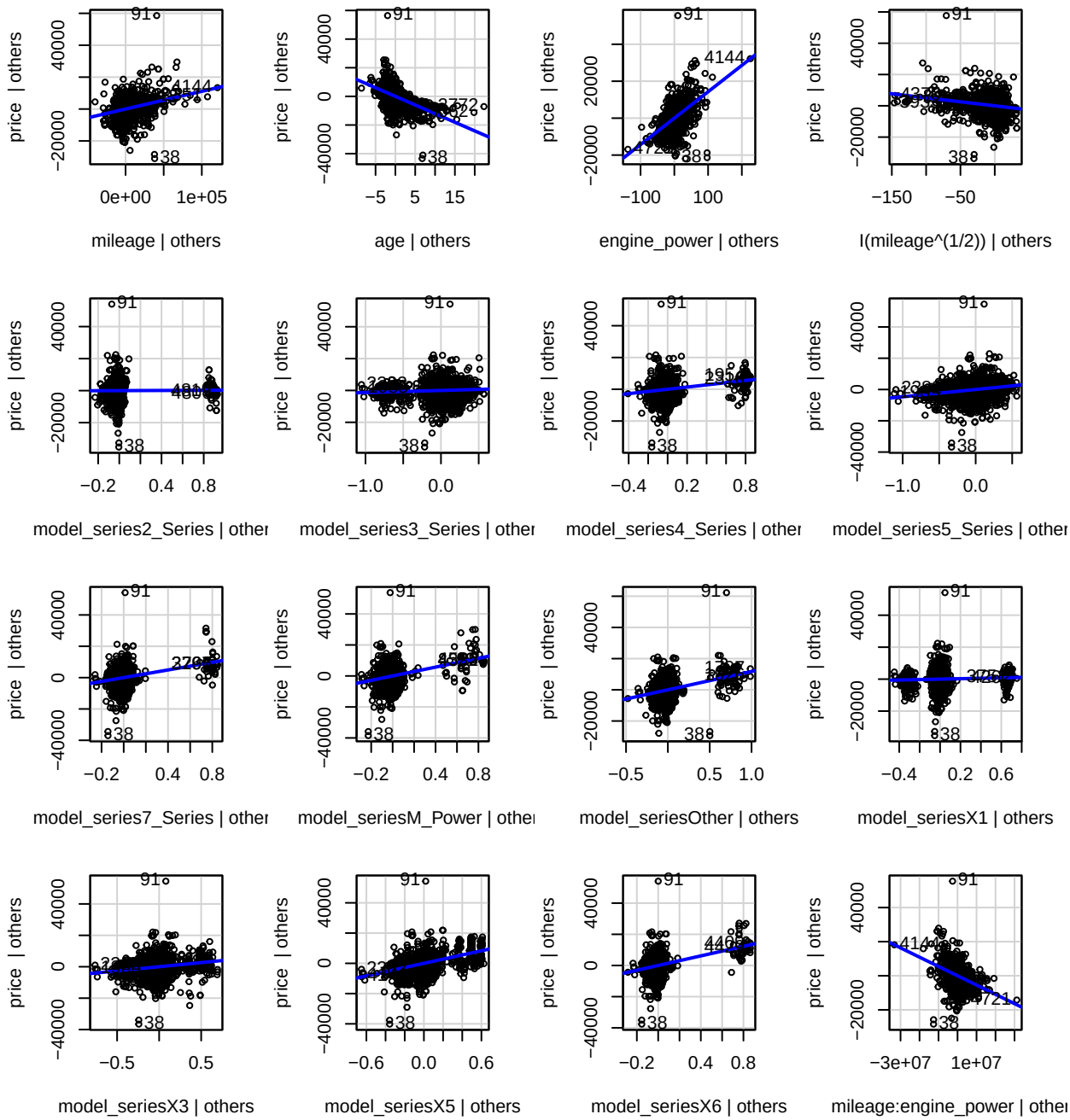
We then investigated if model series could explain a significant amount of variance in auction price with model: $price = \beta_0 + \beta_1 mileage + \beta_2 engine\ power + \beta_3 model\ series + \epsilon$.

```
key_model <- lm(price ~ mileage + age + engine_power + I(mileage^(1/2)) +
    mileage:engine_power + model_series, data = dat_bmw_clean)
summary(key_model)
```

```
##
## Call:
## lm(formula = price ~ mileage + age + engine_power + I(mileage^(1/2)) +
##     mileage:engine_power + model_series, data = dat_bmw_clean)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -35270  -1546     91   1850  54055
```

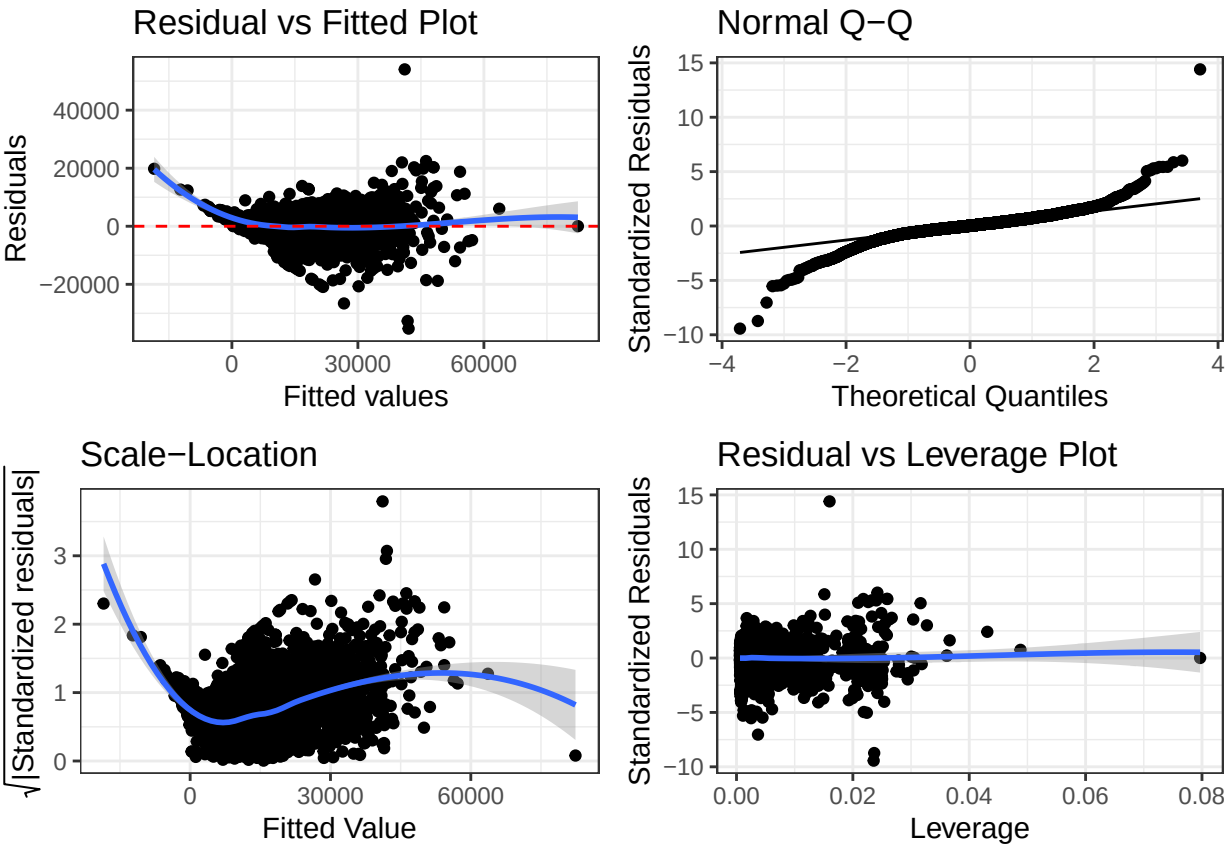
```
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.327e+04  7.997e+02  16.594  < 2e-16 ***
## mileage                 1.109e-01  5.652e-03  19.616  < 2e-16 ***
## age                    -1.206e+03  2.556e+01 -47.171  < 2e-16 ***
## engine_power            1.429e+02  3.656e+00  39.100  < 2e-16 ***
## I(mileage^(1/2))       -5.126e+01  3.498e+00 -14.653  < 2e-16 ***
## model_series2_Series    2.472e+02  5.639e+02   0.438    0.661
## model_series3_Series    1.308e+03  1.915e+02   6.830 9.52e-12 ***
## model_series4_Series    6.900e+03  4.216e+02  16.365  < 2e-16 ***
## model_series5_Series    4.659e+03  2.234e+02  20.854  < 2e-16 ***
## model_series7_Series    1.221e+04  5.813e+02  21.006  < 2e-16 ***
## model_seriesM_Power     1.420e+04  6.450e+02  22.018  < 2e-16 ***
## model_seriesOther       1.177e+04  4.600e+02  25.581  < 2e-16 ***
## model_seriesX1          1.369e+03  2.813e+02   4.868 1.16e-06 ***
## model_seriesX3          5.270e+03  2.559e+02  20.596  < 2e-16 ***
## model_seriesX5          1.383e+04  3.541e+02  39.055  < 2e-16 ***
## model_seriesX6          1.536e+04  6.415e+02  23.939  < 2e-16 ***
## mileage:engine_power -5.453e-04  2.289e-05 -23.825  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3784 on 4821 degrees of freedom
## Multiple R-squared:  0.8111, Adjusted R-squared:  0.8105
## F-statistic:  1294 on 16 and 4821 DF,  p-value: < 2.2e-16
```

```r
avPlots(key_model, layout = c(5, 4))
```




```
diagPlot(key_model)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



Question 3

We calculated the variable "age" to solve the third question: "How does the annual depreciation rate (price decrease per year of age) differ across the top five model keys?"

```
dat_age <- dat_bmw_clean |>
    mutate(age = 2018 - year(as.Date(registration_date, format = "%m/%d/%Y"))) |>
    mutate(is_116 = ifelse(model_key == "116", 1, 0)) |>
    mutate(is_318 = ifelse(model_key == "318", 1, 0)) |>
    mutate(is_X1 = ifelse(model_key == "X1", 1, 0)) |>
    mutate(is_X3 = ifelse(model_key == "X3", 1, 0))

dat_age$age = dat_bmw_clean$age

# determine models with sufficient data (n > 100)
model_candidates <- dat_bmw_clean |>
    group_by(model_key) |>
    summarize(n = n()) |>
    filter(n > 100)
model_candidates
```

```
## # A tibble: 11 x 2
##    model_key     n
##    <chr>     <int>
##  1 116         358
##  2 118         142
##  3 316         235
```

```
##  4 318         569
##  5 320         752
##  6 520         633
##  7 525         184
##  8 530         157
##  9 X1          274
## 10 X3          437
## 11 X5          231
```

```r
# model_candidates$model_key

# narrow down number of models considered
coefs = c()
confintlbs = c()

for (i in seq(1:length(model_candidates$model_key))) {
    temp.df <- dat_age |>
        filter(model_key == model_candidates$model_key[i])
    temp.lm <- lm(log(price) ~ age, data = temp.df)
    coefs = c(coefs, coef(temp.lm)[2])
    confintlbs = c(confintlbs, confint(temp.lm)[2])
}

results <- cbind(coefs, confintlbs, model_candidates$model_key)
results
```

```
##     coefs                 confintlbs
## age "-0.118819880264632"  "-0.141414297461411"  "116"
## age "-0.141452073816491"  "-0.160481629771565"  "118"
## age "-0.148796667047443"  "-0.170617494456091"  "316"
## age "-0.135300507896005"  "-0.149096051756691"  "318"
## age "-0.162027635818915"  "-0.174687855138188"  "320"
## age "-0.18451485362989"   "-0.1926687534329"    "520"
## age "-0.205033387369809"  "-0.223023055608812"  "525"
## age "-0.179154772807124"  "-0.191470718998209"  "530"
## age "-0.0270427628386702" "-0.0748917277565339" "X1"
## age "-0.119133868772023"  "-0.136089017006688"  "X3"
## age "-0.171072537302907"  "-0.178106581981699"  "X5"
```

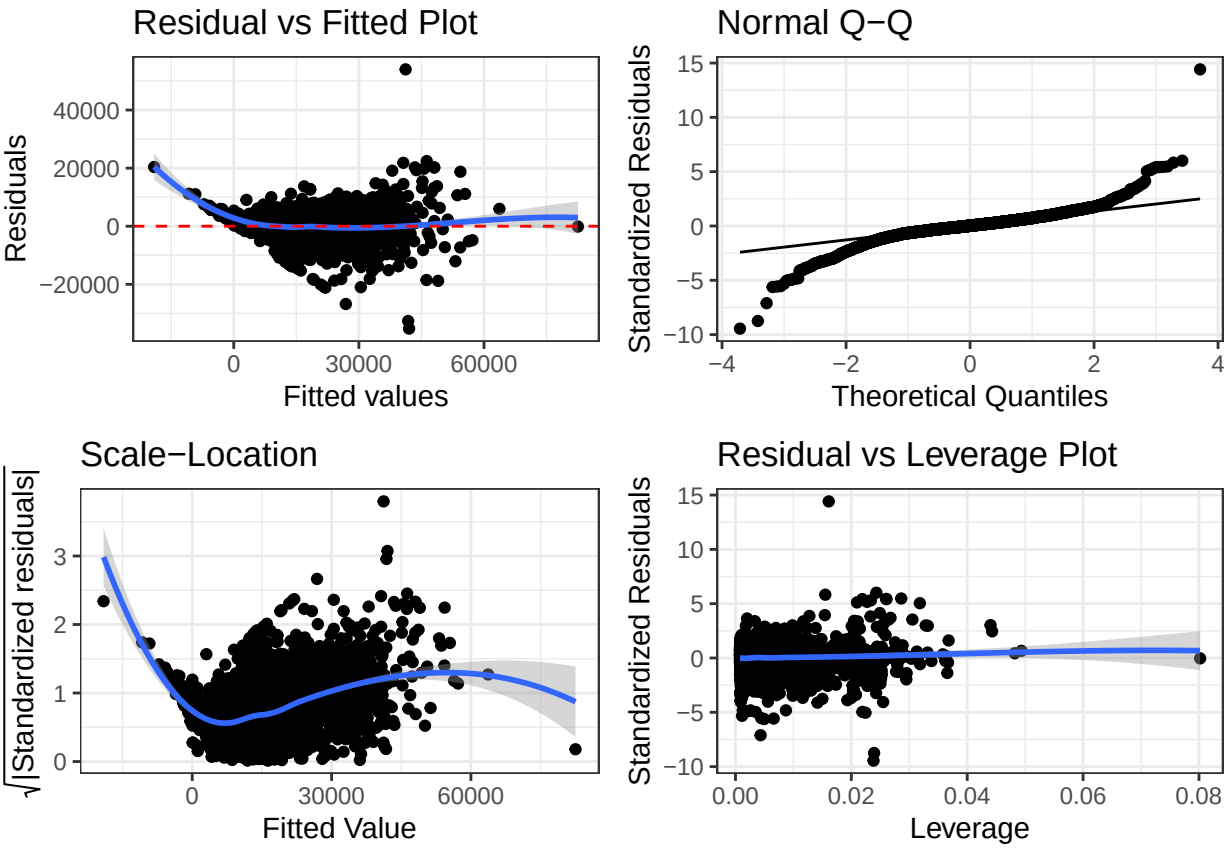
```r
age_model <- lm(formula = price ~ mileage + engine_power + model_series +
    I(mileage^(1/2)) + mileage:engine_power + age + age:is_318 +
    age:is_X1 + age:is_X3 + age:is_116, data = dat_age)
summary(age_model)
```

```
##
## Call:
## lm(formula = price ~ mileage + engine_power + model_series +
##     I(mileage^(1/2)) + mileage:engine_power + age + age:is_318 +
##     age:is_X1 + age:is_X3 + age:is_116, data = dat_age)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -35244  -1526     93   1830  53995
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.303e+04  8.180e+02  15.931  < 2e-16 ***
## mileage                 1.114e-01  5.673e-03  19.638  < 2e-16 ***
## engine_power            1.434e+02  3.659e+00  39.203  < 2e-16 ***
## model_series2_Series    6.254e+02  5.910e+02   1.058  0.29002
## model_series3_Series    1.602e+03  2.727e+02   5.876 4.48e-09 ***
## model_series4_Series    7.262e+03  4.556e+02  15.938  < 2e-16 ***
## model_series5_Series    5.049e+03  2.857e+02  17.675  < 2e-16 ***
## model_series7_Series    1.253e+04  6.032e+02  20.769  < 2e-16 ***
## model_seriesM_Power     1.444e+04  6.615e+02  21.833  < 2e-16 ***
## model_seriesOther       1.210e+04  4.888e+02  24.761  < 2e-16 ***
## model_seriesX1         -1.362e+03  1.068e+03  -1.276  0.20211
## model_seriesX3          6.163e+03  5.447e+02  11.314  < 2e-16 ***
## model_seriesX5          1.415e+04  3.908e+02  36.216  < 2e-16 ***
## model_seriesX6          1.566e+04  6.602e+02  23.713  < 2e-16 ***
## I(mileage^(1/2))       -5.223e+01  3.515e+00 -14.859  < 2e-16 ***
## age                    -1.222e+03  2.709e+01 -45.099  < 2e-16 ***
## mileage:engine_power   -5.381e-04  2.295e-05 -23.445  < 2e-16 ***
## age:is_318              7.982e+01  3.052e+01   2.615  0.00895 **
## age:is_X1               6.175e+02  1.976e+02   3.124  0.00179 **
## age:is_X3              -8.949e+01  8.263e+01  -1.083  0.27887
## age:is_116              1.373e+02  5.692e+01   2.412  0.01589 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3776 on 4817 degrees of freedom
## Multiple R-squared:  0.812,  Adjusted R-squared:  0.8112
## F-statistic:  1040 on 20 and 4817 DF,  p-value: < 2.2e-16
```

```
diagPlot(age_model)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



Final Model

As we found before, age significantly associated with price, therefore we included it into our final model: $price = \beta_0 + \beta_1 mileage + \beta_2 engine\ power + \beta_3 model\ series + \beta 4\sqrt{mileage} + \beta_5 mileage \times engine\ power + \beta_6 age + \epsilon$.

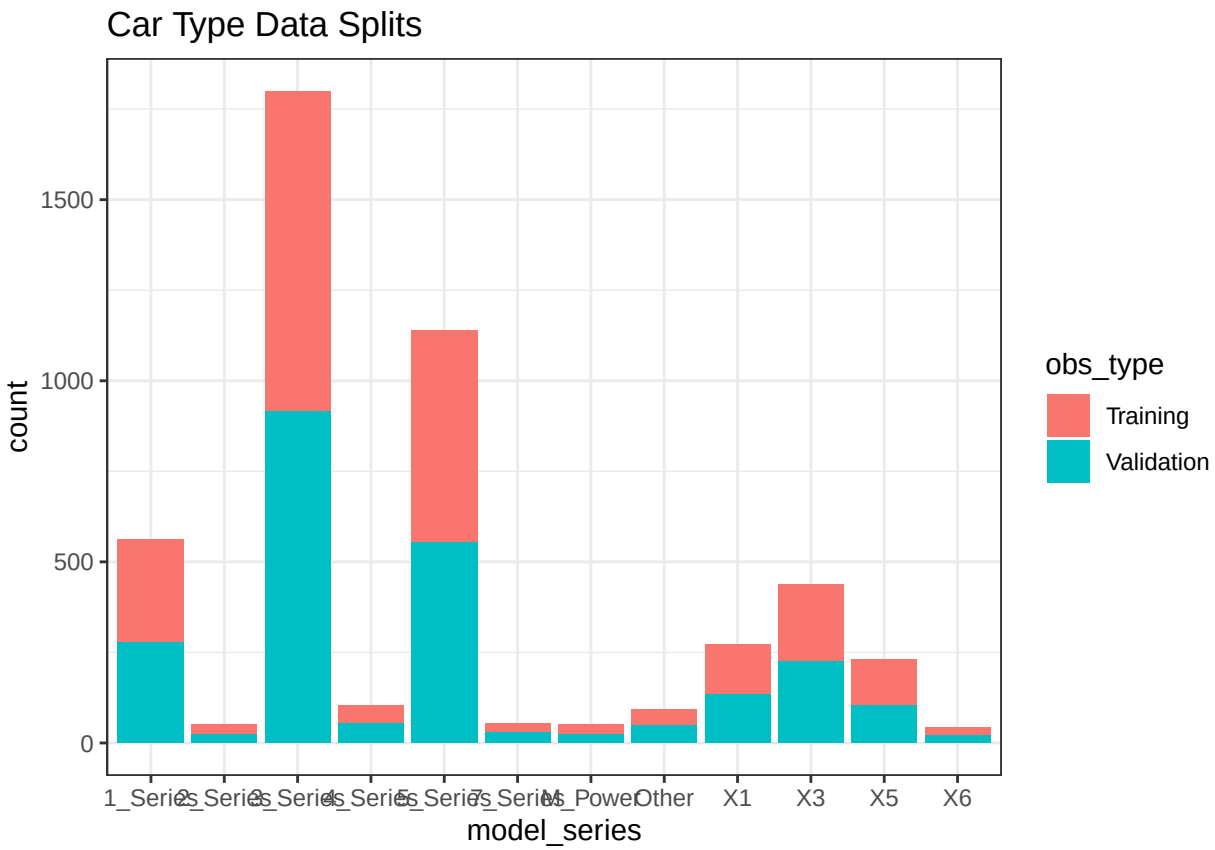
Modelling and Analysis for the Training Data Set

Split Training and Testing Data

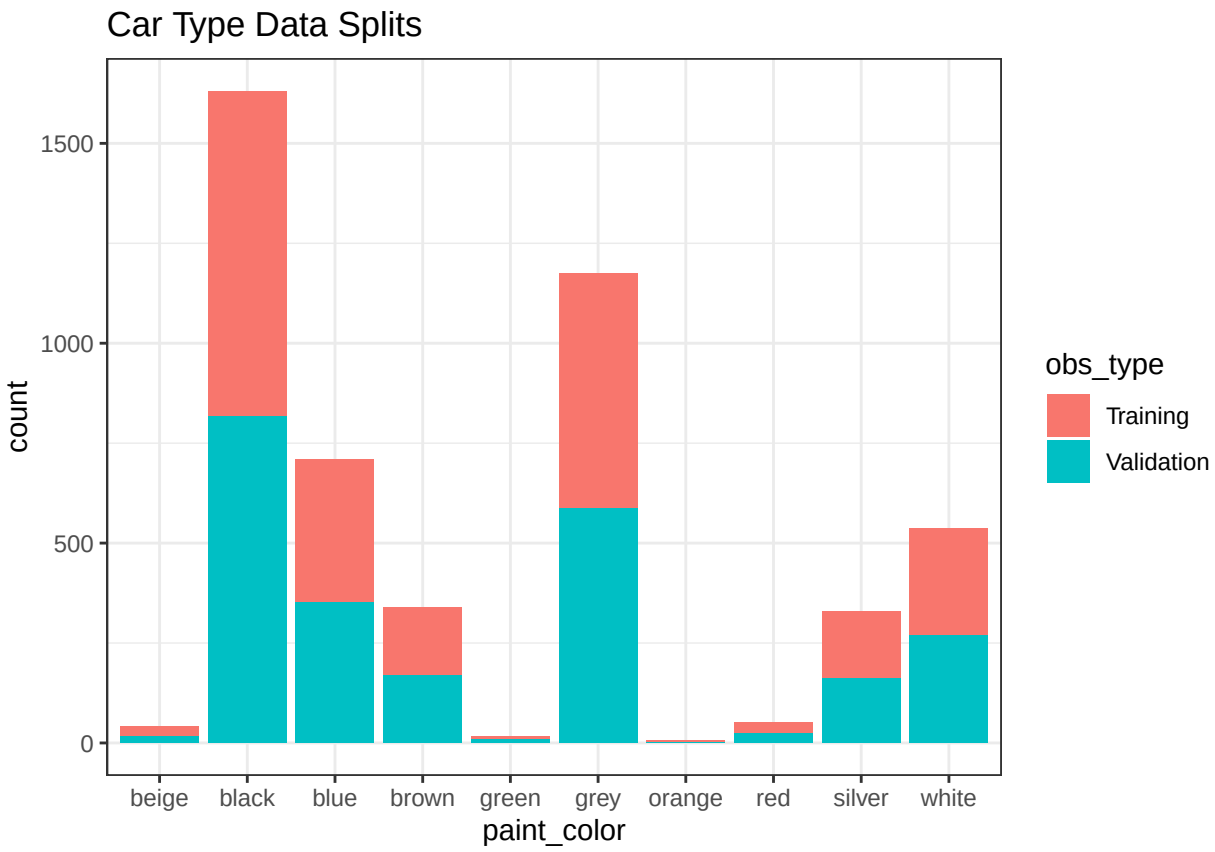
```
train_index <- which(dat_bmw_clean$obs_type == "Training")
test_index <- which(dat_bmw_clean$obs_type == "Validation")
dat_bmw_train <- dat_bmw_clean[train_index, ]
dat_bmw_test <- dat_bmw_clean[test_index, ]
```

Data Overview of the Training and Testing Data

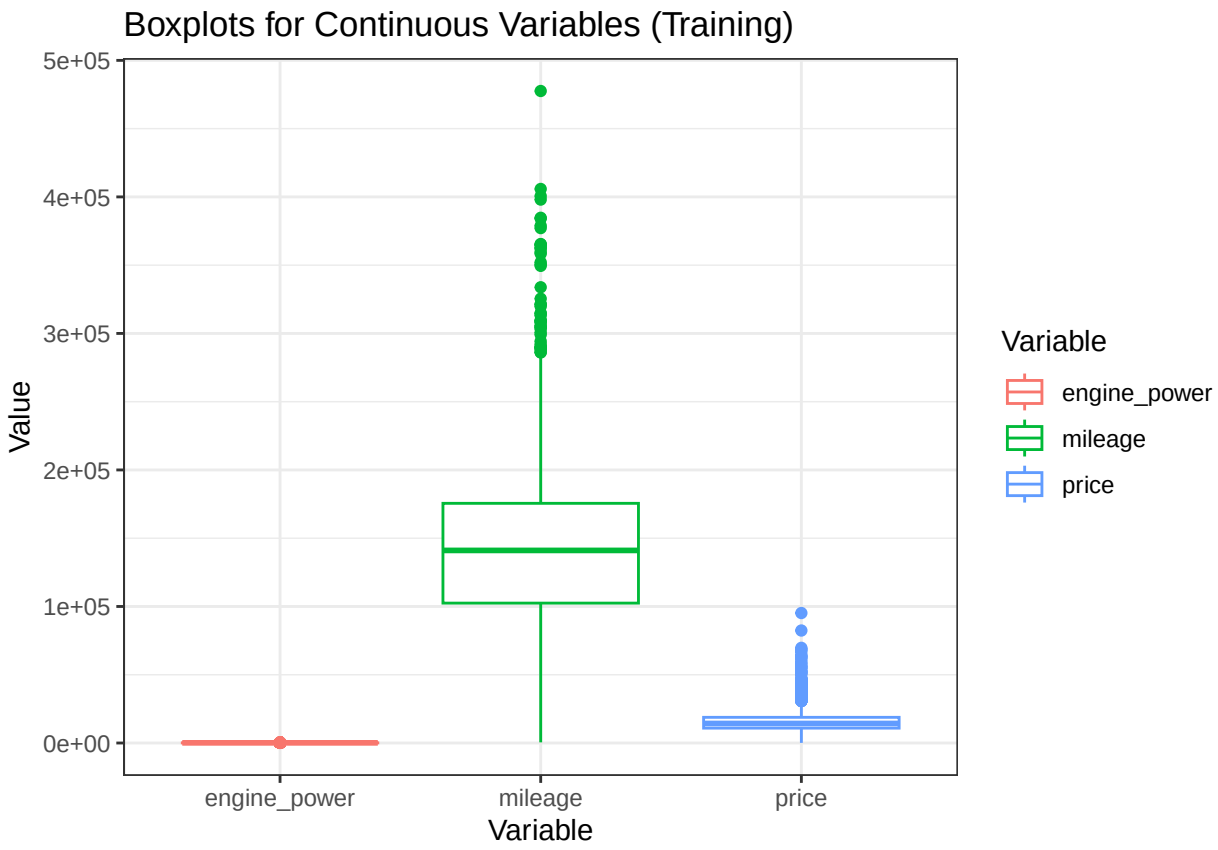
```
dat_bmw_clean |>
    ggplot(aes(x = model_series, fill = obs_type)) + geom_bar() +
    labs(title = "Car Type Data Splits") + theme(axis.text.x = element_text(angle =
     ↪  30)) +
    theme_bw()
```



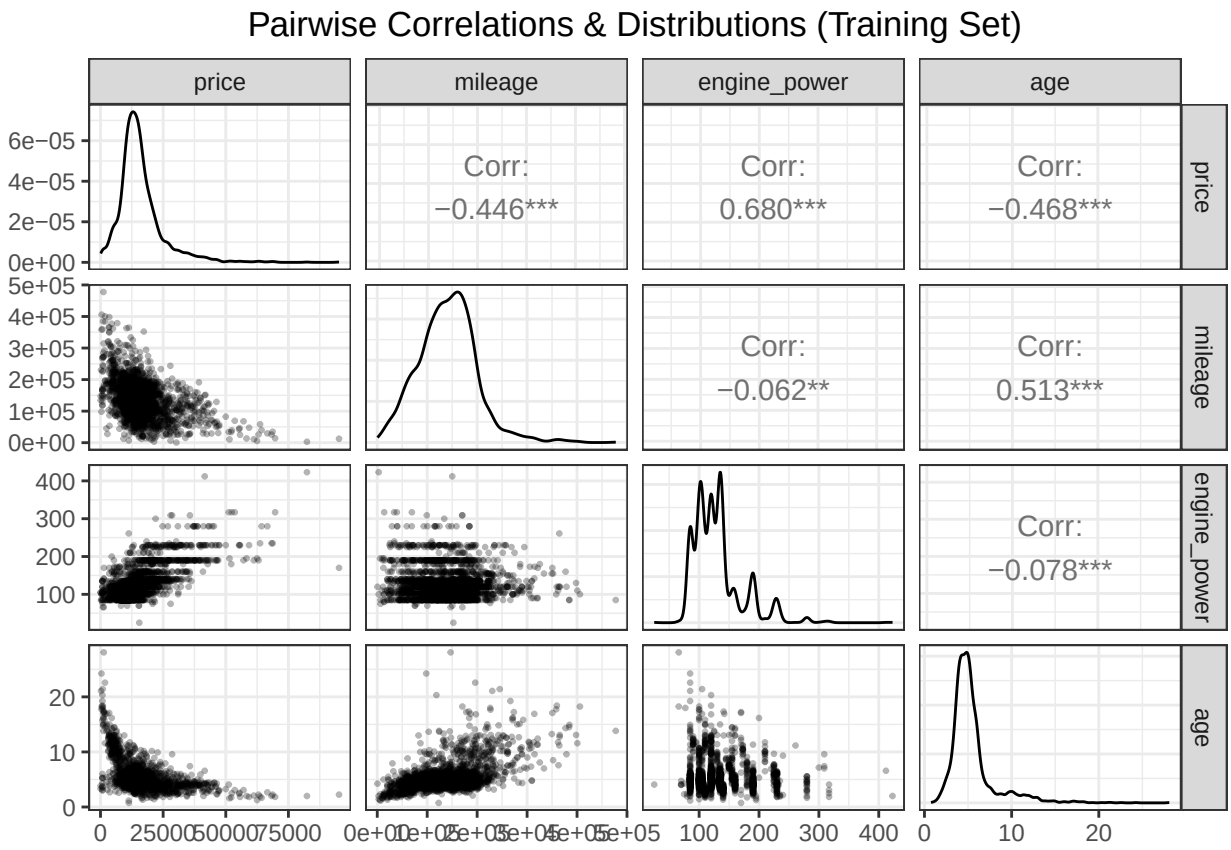

```
dat_bmw_clean |>
    ggplot(aes(x = paint_color, fill = obs_type)) + geom_bar() +
    labs(title = "Car Type Data Splits") + theme(axis.text.x = element_text(angle =
     ↪  30)) +
    theme_bw()
```




```
dat_bmw_clean[, c("price", "mileage", "engine_power", "obs_type")] |>
    filter(obs_type == "Training") |>
    dplyr::select(price, mileage, engine_power) |>
    pivot_longer(everything(), values_to = "Value", names_to = "Variable") |>
    ggplot() + geom_boxplot(aes(x = Variable, y = Value, color = Variable)) +
    labs(title = "Boxplots for Continuous Variables (Training)") +
    theme_bw()
```




```
ggpairs(dat_bmw_train[, c("price", "mileage", "engine_power",
    "age")], title = "Pairwise Correlations & Distributions (Training Set)",
    upper = list(continuous = wrap("cor", size = 4)), lower = list(continuous =
     ↪  wrap("points",
        alpha = 0.3, size = 0.5)), diag = list(continuous = wrap("densityDiag"))) +
    theme_bw() + theme(plot.title = element_text(hjust = 0.5))
```



Final Model Fit on Training Data

The final model we fit is $price = \beta_0 + \beta_1 mileage + \beta_2 engine\ power + \beta_3 model\ series + \beta_4 age + \beta 5\sqrt{mileage} + \beta_6 mileage \times engine\ power + \epsilon$.

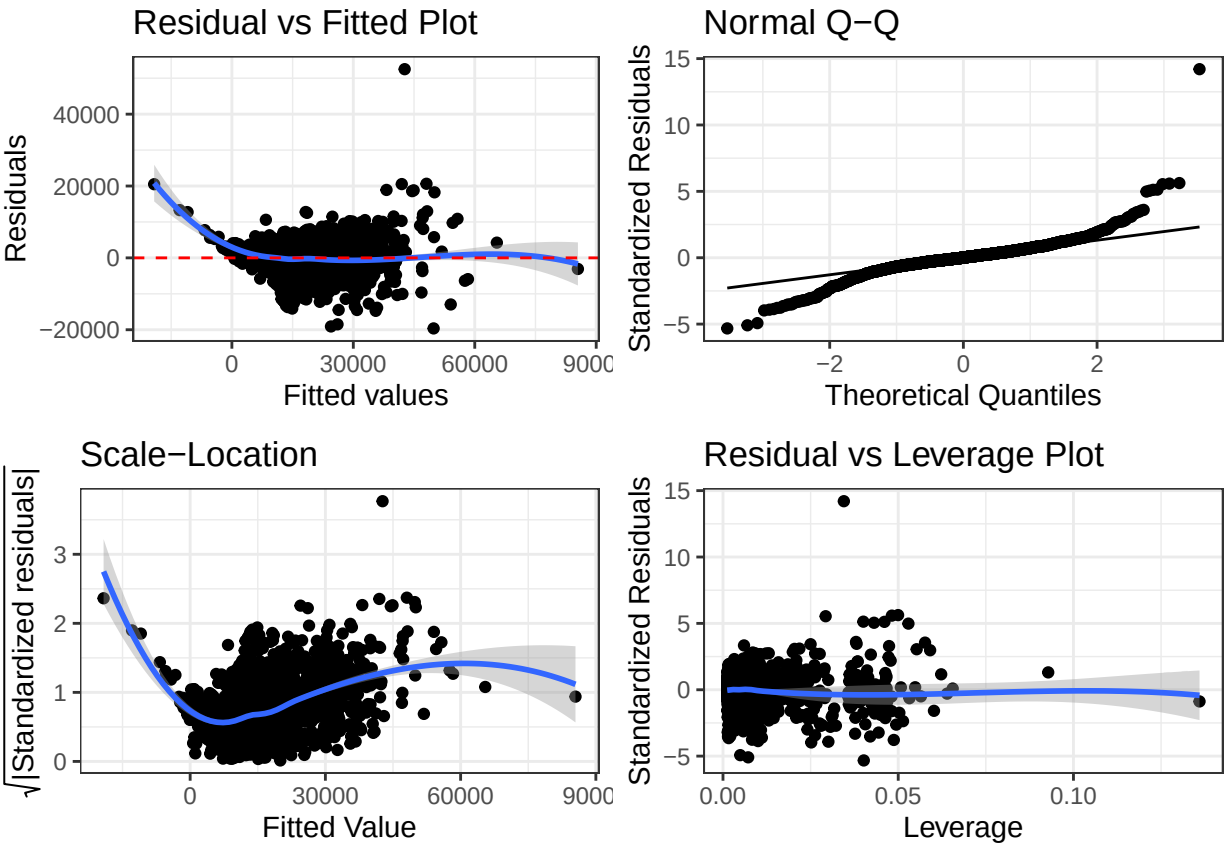
```
final_model_train <- lm(price ~ mileage + engine_power + model_series +
    age + I(mileage^(1/2)) + mileage:engine_power, data = dat_bmw_train)
summary(final_model_train)
```

```
##
## Call:
## lm(formula = price ~ mileage + engine_power + model_series +
##     age + I(mileage^(1/2)) + mileage:engine_power, data = dat_bmw_train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -19635  -1583     57   1716  52516
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.423e+04  1.173e+03  12.128  < 2e-16 ***
## mileage                 1.289e-01  7.851e-03  16.419  < 2e-16 ***
## engine_power            1.504e+02  5.023e+00  29.945  < 2e-16 ***
## model_series2_Series -3.297e+02  7.598e+02  -0.434 0.664418
## model_series3_Series  1.097e+03  2.676e+02   4.098 4.31e-05 ***
## model_series4_Series  6.372e+03  6.026e+02  10.574  < 2e-16 ***
## model_series5_Series  4.348e+03  3.117e+02  13.952  < 2e-16 ***
```

```
## model_series7_Series  1.281e+04  8.726e+02  14.677  < 2e-16 ***
## model_seriesM_Power   1.372e+04  9.143e+02  15.008  < 2e-16 ***
## model_seriesOther     1.198e+04  6.530e+02  18.343  < 2e-16 ***
## model_seriesX1        1.391e+03  3.909e+02   3.559 0.000379 ***
## model_seriesX3        5.376e+03  3.598e+02  14.942  < 2e-16 ***
## model_seriesX5        1.302e+04  4.848e+02  26.852  < 2e-16 ***
## model_seriesX6        1.409e+04  8.929e+02  15.775  < 2e-16 ***
## age                  -1.235e+03  3.590e+01 -34.399  < 2e-16 ***
## I(mileage^(1/2))     -5.973e+01  5.061e+00 -11.803  < 2e-16 ***
## mileage:engine_power -5.994e-04  3.097e-05 -19.353  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3763 on 2414 degrees of freedom
## Multiple R-squared:  0.8231, Adjusted R-squared:  0.8219
## F-statistic:   702 on 16 and 2414 DF,  p-value: < 2.2e-16
```

```
diagPlot(final_model_train)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



Lasso Regression

We performed Lasso regression for the variable selection.

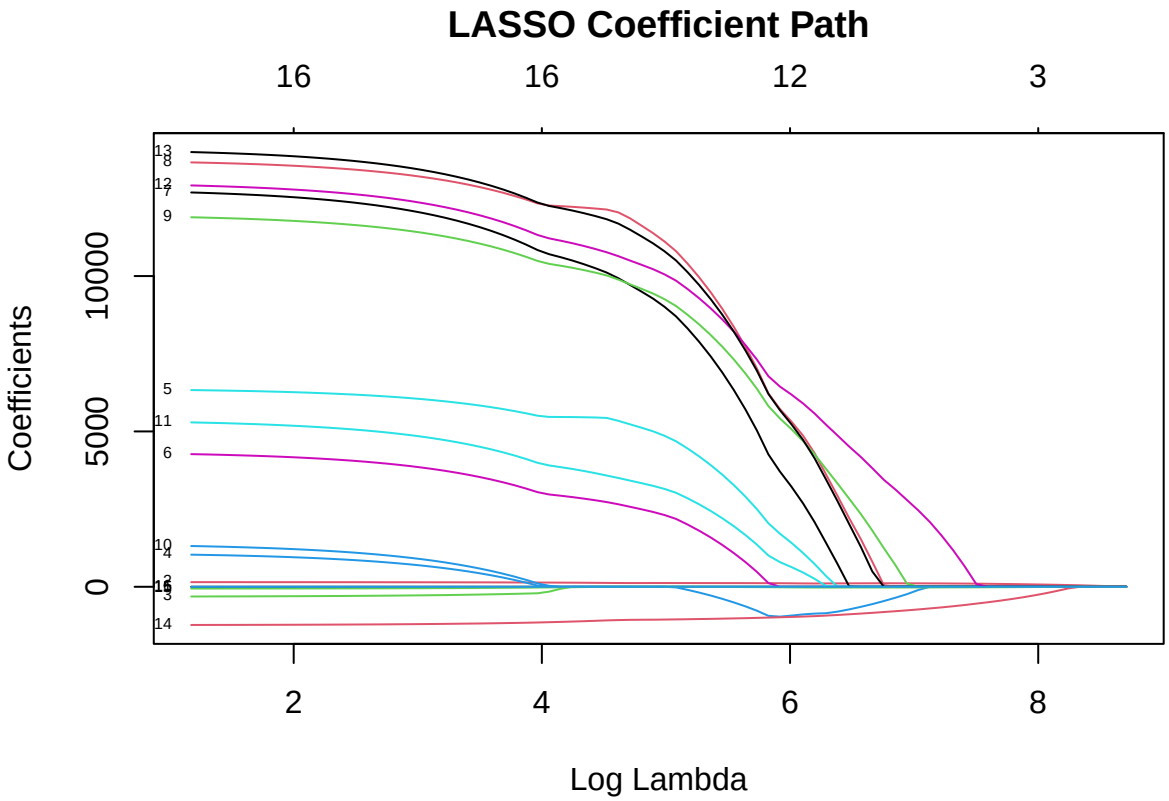
```r
# Create a model matrix
X <- model.matrix(price ~ mileage + engine_power + model_series +
    age + I(mileage^(1/2)) + mileage:engine_power, data = dat_bmw_train)[,
    -1]

# Create the response variable
Y <- dat_bmw_train$price

# set seed for reproductivity
set.seed(20250408)

# Fit the LASSO regression model using glmnet with alpha =
# 1
lasso_model <- glmnet(x = X, y = Y, alpha = 1)

# The x-axis shows log(lambda) and each line corresponds to
# a coefficient.
plot(lasso_model, xvar = "lambda", label = TRUE)
title("LASSO Coefficient Path", line = 2.5)
```



We used cross-validation to determine the lambda that minimizes the mean squared error.

```r
cv_model <- cv.glmnet(x = X, y = Y, alpha = 1)
best_lambda <- cv_model$lambda.min

cat("Best Lambda - LASSO:", best_lambda)
```

```
## Best Lambda - LASSO: 3.234687
```

The coefficient for the interaction was close to 0 (i.e., $\beta_{mileage \times engine\ power} = -5.87 \times 10^{-4}$). Therefore, we used partial F-test to investigate if we can remove this interaction term.

```
lasso_coef <- coef(lasso_model, s = best_lambda)
print(lasso_coef)
```

```
## 17 x 1 sparse Matrix of class "dgCMatrix"
##                                    s1
## (Intercept)           1.397773e+04
## mileage               1.233594e-01
## engine_power          1.491882e+02
## model_series2_Series -3.124235e+02
## model_series3_Series  1.033387e+03
## model_series4_Series  6.331205e+03
## model_series5_Series  4.269069e+03
## model_series7_Series  1.269119e+04
## model_seriesM_Power   1.365871e+04
## model_seriesOther     1.189258e+04
## model_seriesX1        1.311942e+03
## model_seriesX3        5.290716e+03
## model_seriesX5        1.291696e+04
## model_seriesX6        1.399076e+04
## age                  -1.228902e+03
## I(mileage^(1/2))     -5.700250e+01
## mileage:engine_power -5.870496e-04
```

```
reduced_model <- lm(price ~ mileage + engine_power + model_series +
    age + I(mileage^(1/2)), data = dat_bmw_train)
anova(final_model_train, reduced_model)
```

```
## Analysis of Variance Table
##
## Model 1: price ~ mileage + engine_power + model_series + age + I(mileage^(1/2)) +
##     mileage:engine_power
## Model 2: price ~ mileage + engine_power + model_series + age + I(mileage^(1/2))
##   Res.Df        RSS Df   Sum of Sq      F    Pr(>F)
## 1   2414 3.4175e+10
## 2   2415 3.9477e+10 -1 -5302067949 374.52 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Prediction

With the testing data, We used the coefficients from Lasso regression to evaluate the prediction performance of our final model: $price = \beta_0 + \beta_1 mileage + \beta_2 engine\ power + \beta_3 model\ series + \beta_4 age + \beta 5\sqrt{mileage} + \beta_6 mileage \times engine\ power + \epsilon$.

```
newX <- model.matrix(price ~ mileage + engine_power + model_series +
    age + I(mileage^(1/2)) + mileage:engine_power, data = dat_bmw_test)[,
    -1]
```

```r
best_model <- glmnet(x = X, y = Y, alpha = 1, lambda = best_lambda)

pred_values <- predict(best_model, newx = newX)
obs_values <- dat_bmw_test$price

RMSE <- RMSE(pred_values, obs_values)
MAE <- MAE(pred_values, obs_values)
MAPE <- MAPE(pred_values, obs_values)

rbind(RMSE, MAE, MAPE)
```

```
##              [,1]
## RMSE 3826.6398506
## MAE  2538.6284046
## MAPE    0.6997168
```

```r
summary(dat_bmw_test$price)
```

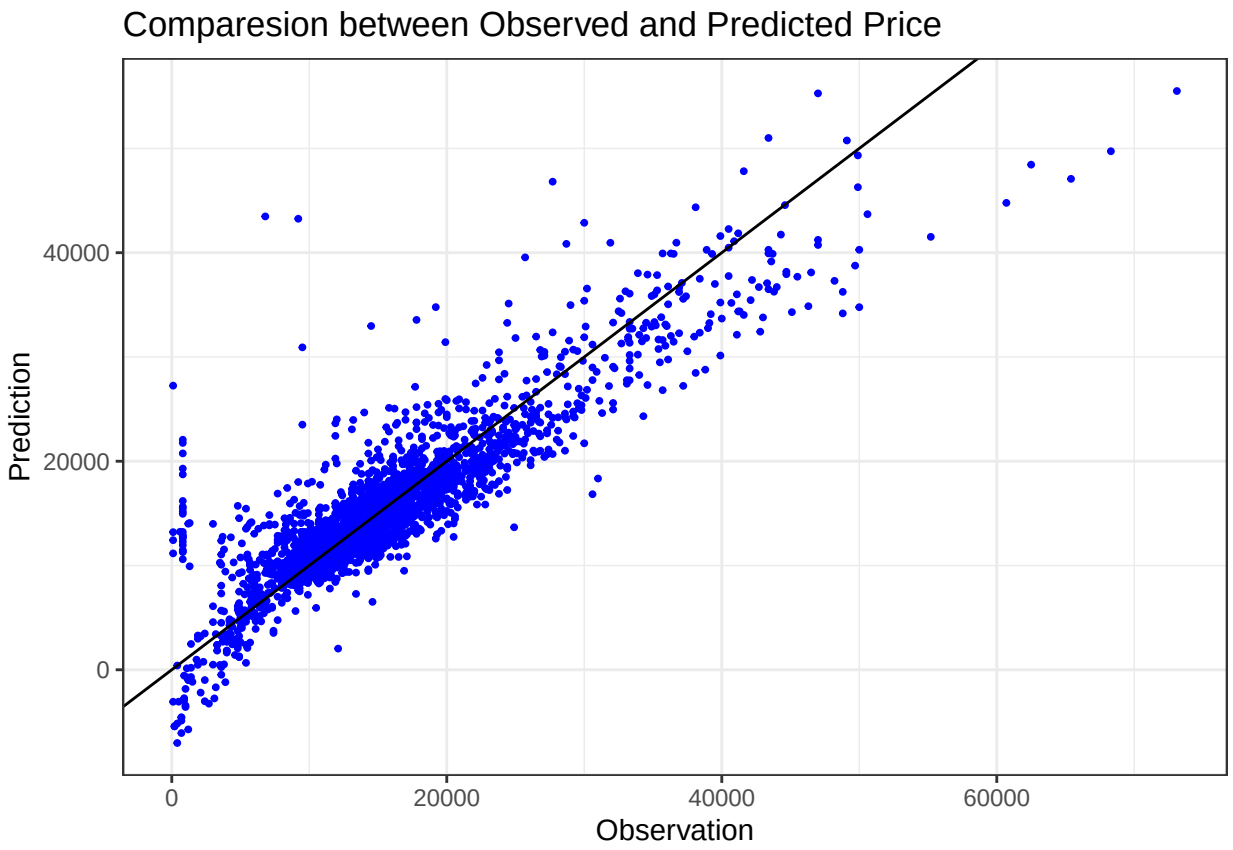
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     100   10800   14000   15601   18600   73100
```

```r
summary(dat_bmw_clean$price)
```

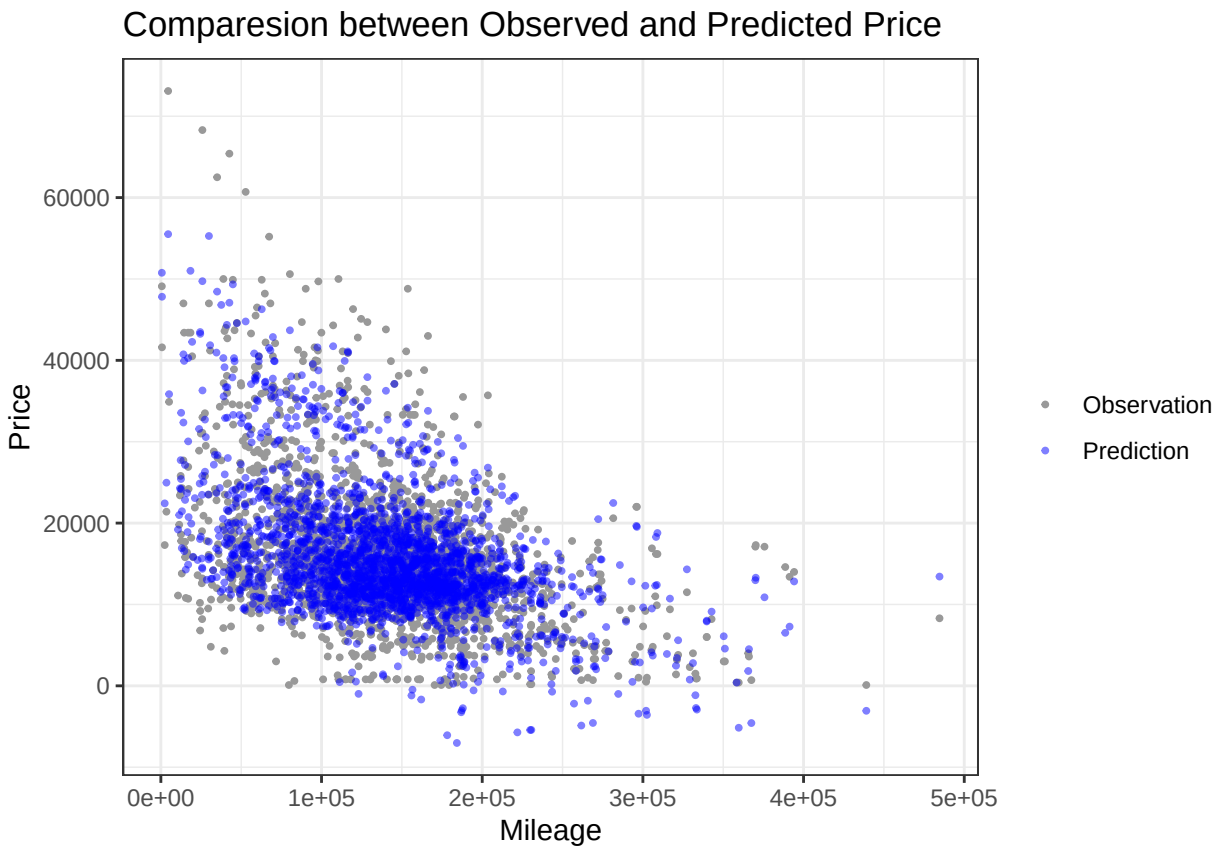
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     100   10800   14200   15760   18600   95200
```

```r
dat_bmw_test$pred <- pred_values

ggplot(data = dat_bmw_test, mapping = aes(x = price, y = pred)) +
    geom_point(size = 0.7, color = "blue") + labs(title = "Comparesion between Observed
     ↪  and Predicted Price") +
    xlab("Observation") + ylab("Prediction") + geom_abline(slope = 1,
    intercept = 0) + theme_bw()
```




```
ggplot() + geom_point(data = dat_bmw_test, aes(x = mileage, y = price,
    color = "Observation"), size = 0.7) + geom_point(data = dat_bmw_test,
    aes(x = mileage, y = pred, color = "Prediction"), alpha = 0.5,
    size = 0.7) + labs(title = "Comparesion between Observed and Predicted Price") +
    xlab("Mileage") + ylab("Price") + scale_color_manual(name = element_blank(),
    values = c(Observation = "grey60", Prediction = "blue")) +
    theme_bw()
```



Data and Codes Availability

All data and codes could be found in our GitHub repository: “https://github.com/mengyaoww/MA575_Lab2_GroupA.git”.