

# 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

<b>Data</b>	<b>2</b>
Data Overview . . . . .	2
Missing data and implausible value handling . . . . .	4
<b>Modelling and Analysis for the Full Data Set</b>	<b>6</b>
Question 1 . . . . .	6
Question 2 . . . . .	7
Model Refinement . . . . .	11
Question 3 . . . . .	15
Final Model . . . . .	18
<b>Modelling and Analysis for the Training Data Set</b>	<b>20</b>
Split Training and Testing Data . . . . .	20
Data Overview of the Training and Testing Data . . . . .	21
Final Model Fit on Training Data . . . . .	23
Lasso Regression . . . . .	25
Prediction . . . . .	27
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

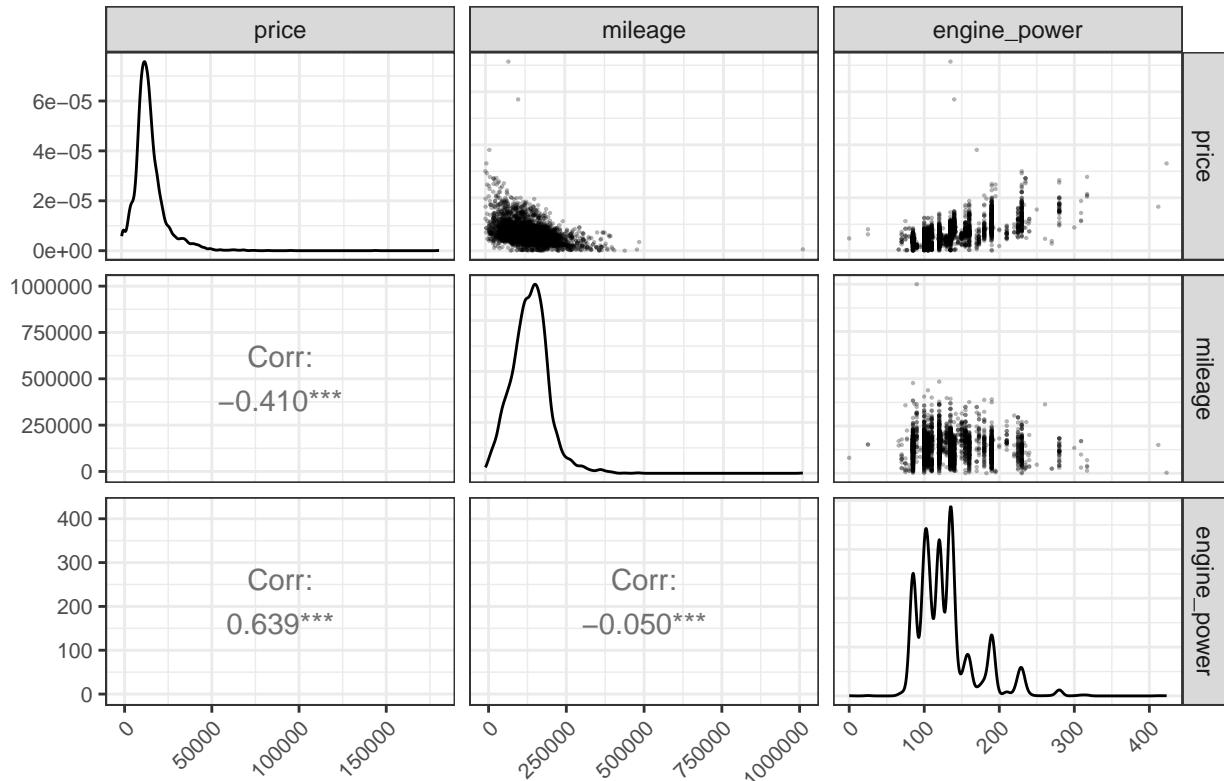
### Data Overview

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))
```

## Distribution and Correlation between Price, Mileage, and Engine Power



Then we investigated the distribution of price, the relationship between price and registration 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"))

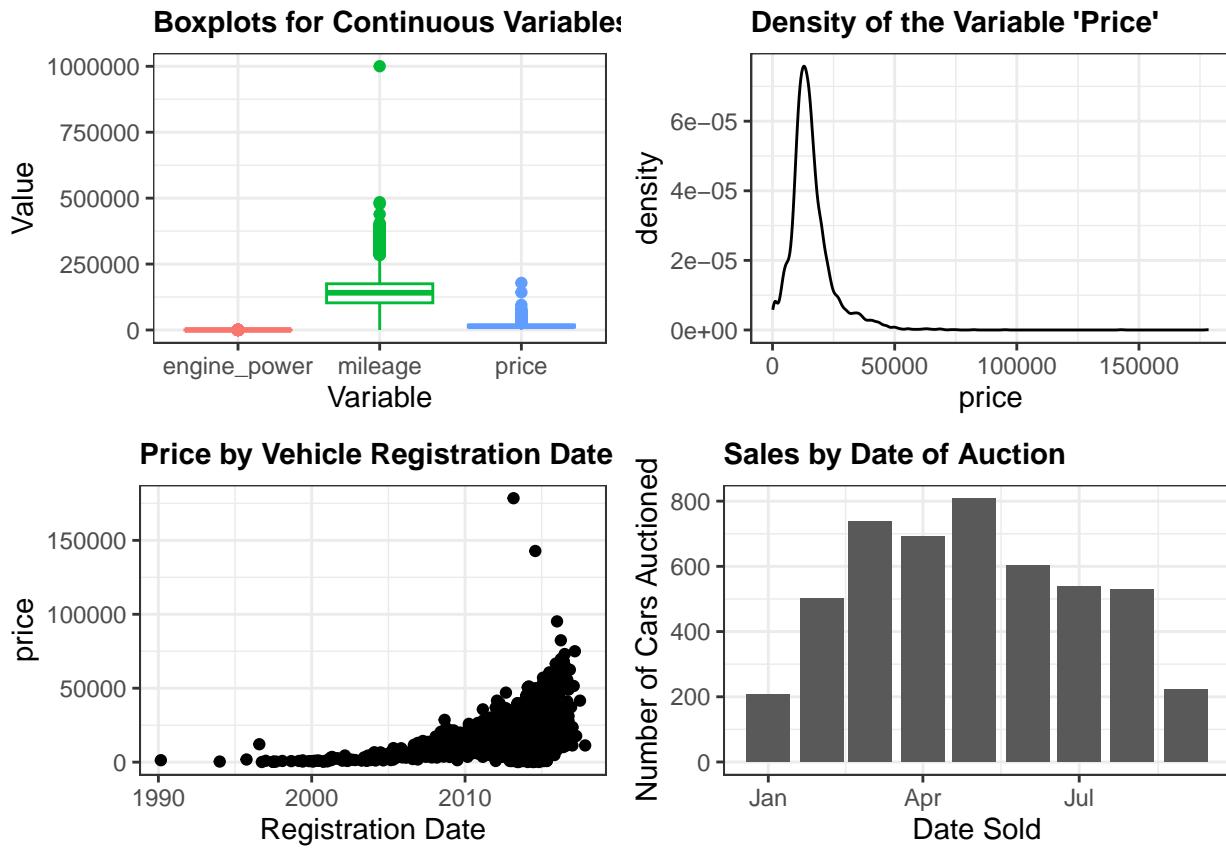
p_sale_Date <- dat_bmw |>
  ggplot(aes(x = as.Date(sold_at, format = "%m/%d/%Y"))) +
  geom_bar() + labs(title = "Sales by Date of Auction") + xlab("Date Sold") +
  ylab("Number of Cars Auctioned") + theme_bw() + theme(plot.title =
  ↪ element_text(size = 11,
  face = "bold"))
  
```

```

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.

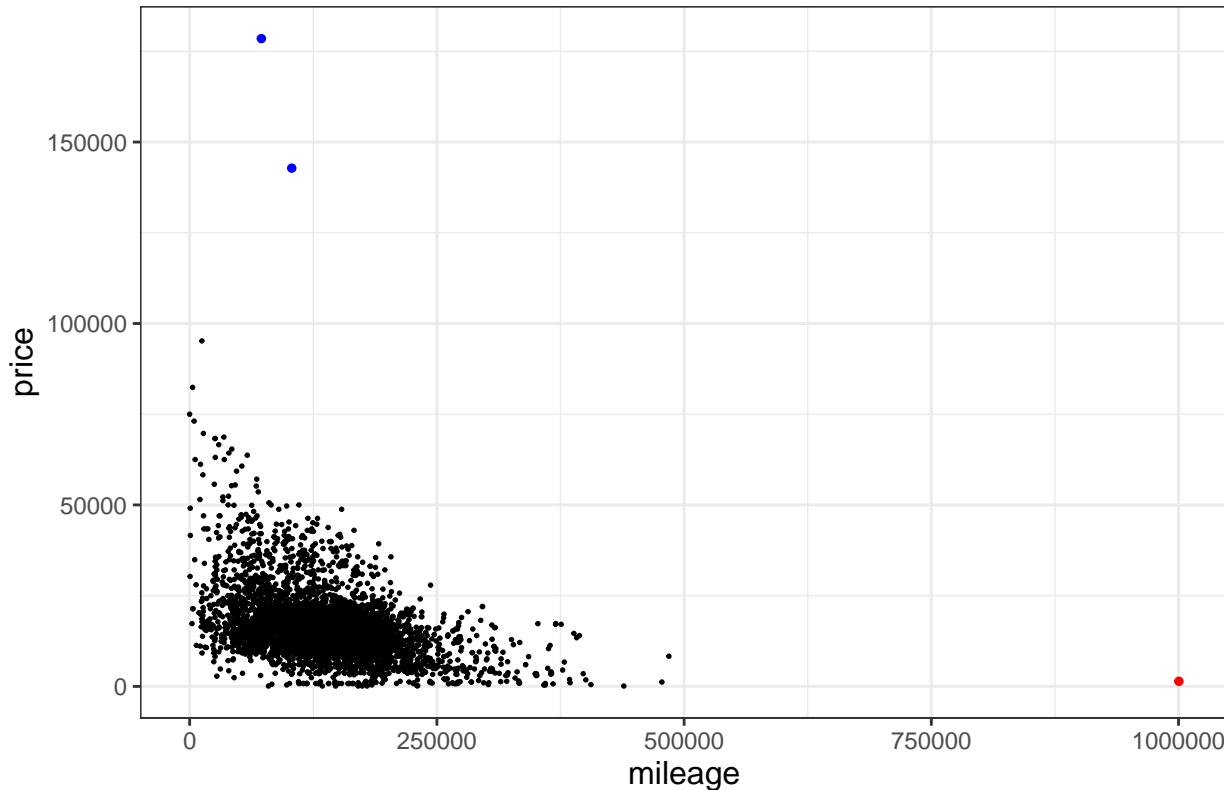
```

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

```

## Scatter plot between price and mileage



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"))

```

## 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:  $price = \beta_0 + \beta_1 \text{mileage} + \beta_2 \text{engine power} + \epsilon$ .

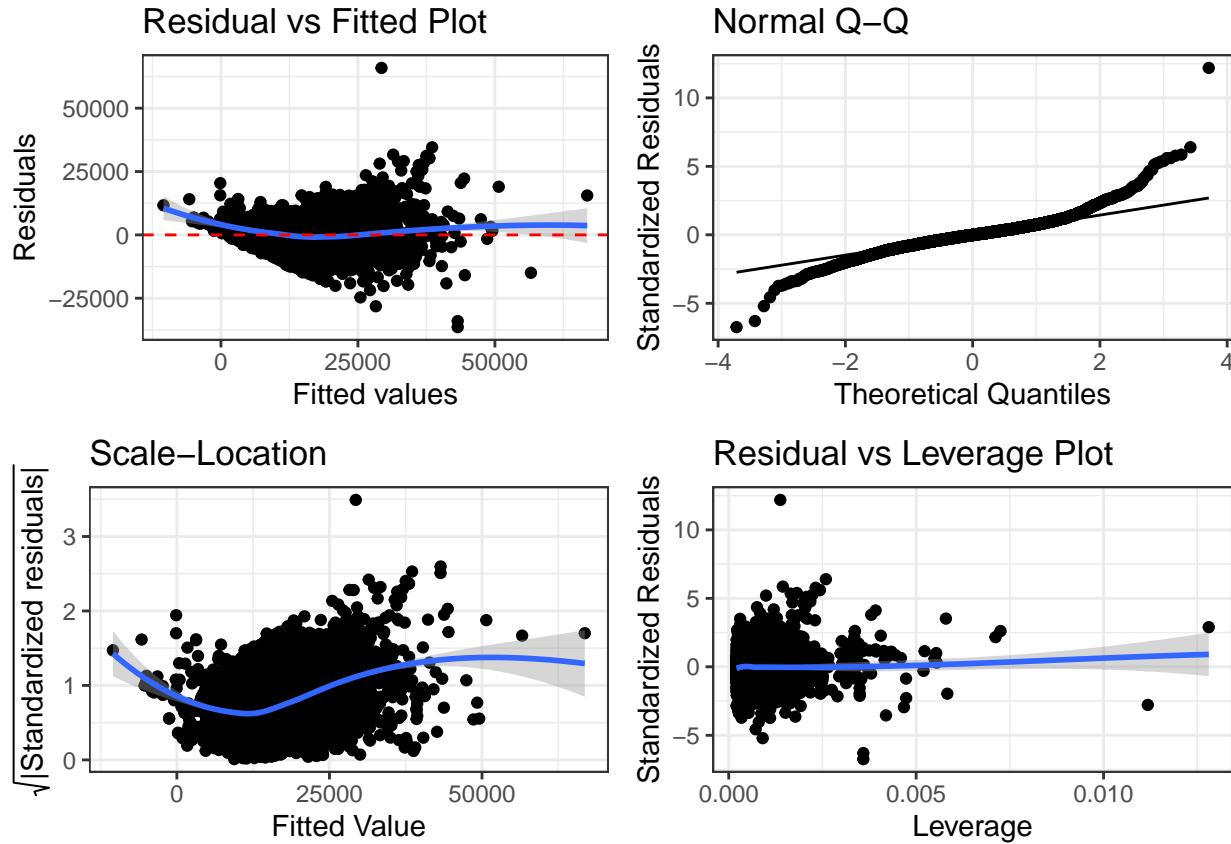
```
baseline_model <- lm(price ~ mileage + engine_power, data = dat_bmw_clean)
summary(baseline_model)
```

```

##
## 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'
```



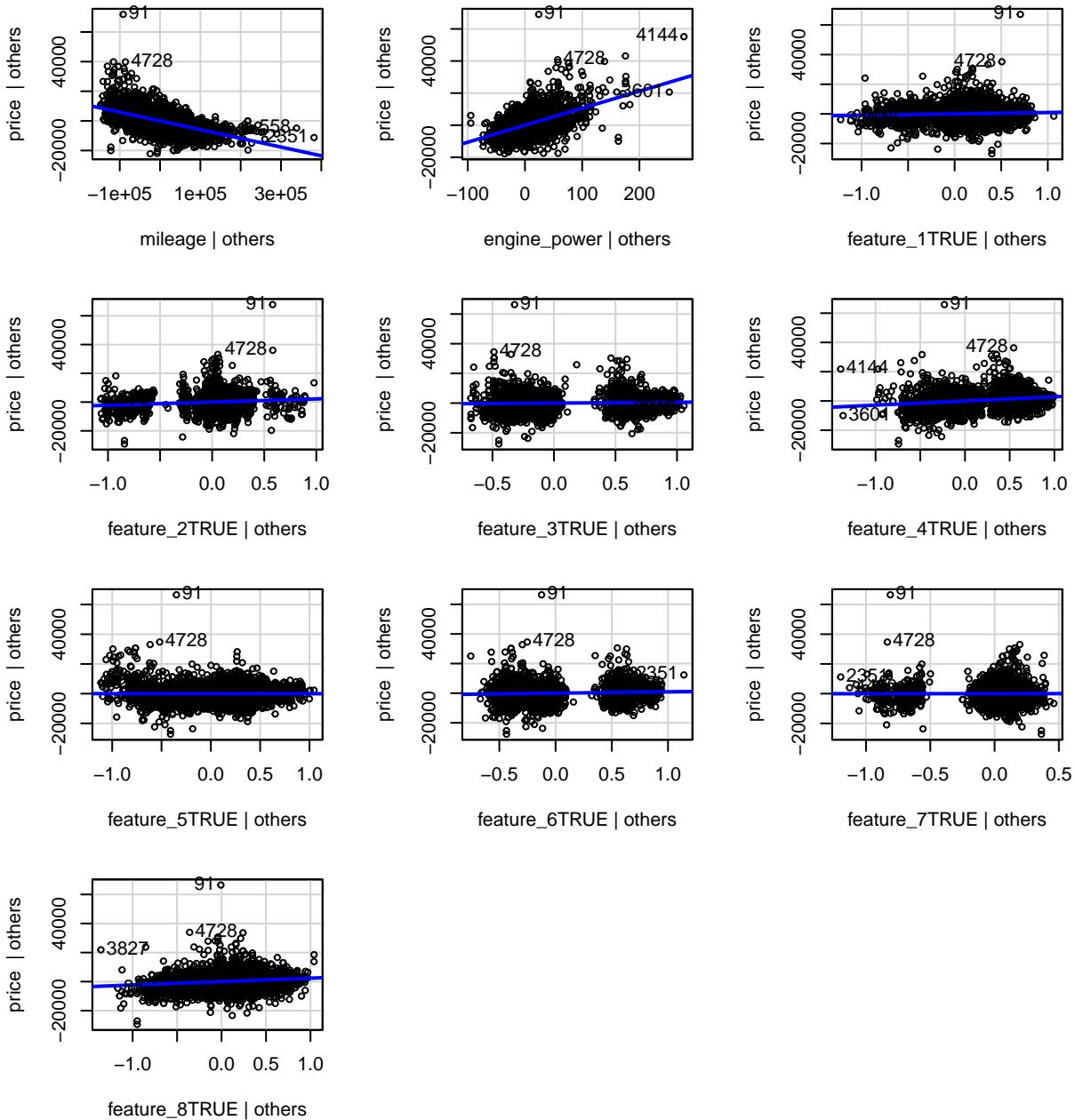
## Question 2

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))
```

## Added-Variable Plots



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 + engine_power + model_series,
  data = dat_bmw_clean)
summary(key_model)
```

```
##
## Call:
## lm(formula = price ~ mileage + engine_power + model_series, data = dat_bmw_clean)
##
## Residuals:
##    Min     1Q Median     3Q    Max 
## -35607 -2007   266  2432 60188
```

```

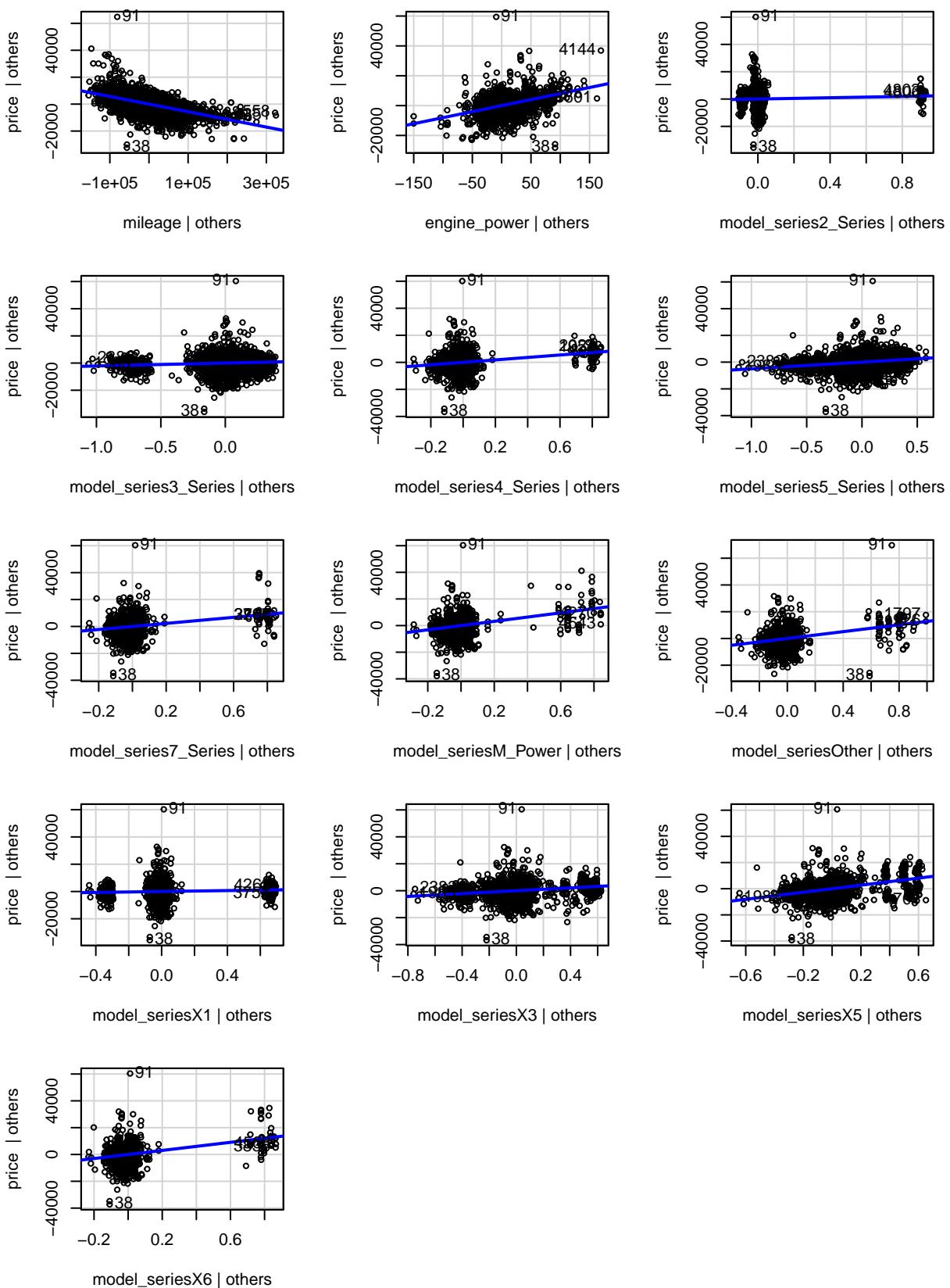
## 
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)          9.240e+03  3.521e+02  26.243 < 2e-16 ***
## mileage            -5.681e-02  1.225e-03 -46.380 < 2e-16 ***
## engine_power        8.080e+01  2.543e+00  31.770 < 2e-16 ***
## model_series2_Series 2.389e+03  6.975e+02   3.426 0.000618 *** 
## model_series3_Series 2.169e+03  2.377e+02   9.126 < 2e-16 *** 
## model_series4_Series 9.068e+03  5.249e+02  17.275 < 2e-16 *** 
## model_series5_Series 4.992e+03  2.798e+02  17.841 < 2e-16 *** 
## model_series7_Series 1.128e+04  7.279e+02  15.495 < 2e-16 *** 
## model_seriesM_Power  1.596e+04  7.991e+02  19.967 < 2e-16 *** 
## model_series0Other   1.274e+04  5.736e+02  22.213 < 2e-16 *** 
## model_seriesX1       1.742e+03  3.520e+02   4.949 7.73e-07 *** 
## model_seriesX3       5.334e+03  3.201e+02  16.661 < 2e-16 *** 
## model_seriesX5       1.349e+04  4.427e+02  30.482 < 2e-16 *** 
## model_seriesX6       1.508e+04  8.018e+02  18.804 < 2e-16 *** 
## ---                
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

## 
## Residual standard error: 4744 on 4824 degrees of freedom
## Multiple R-squared:  0.7028, Adjusted R-squared:  0.702 
## F-statistic: 877.4 on 13 and 4824 DF,  p-value: < 2.2e-16

```

```
avPlots(key_model, layout = c(5, 3))
```

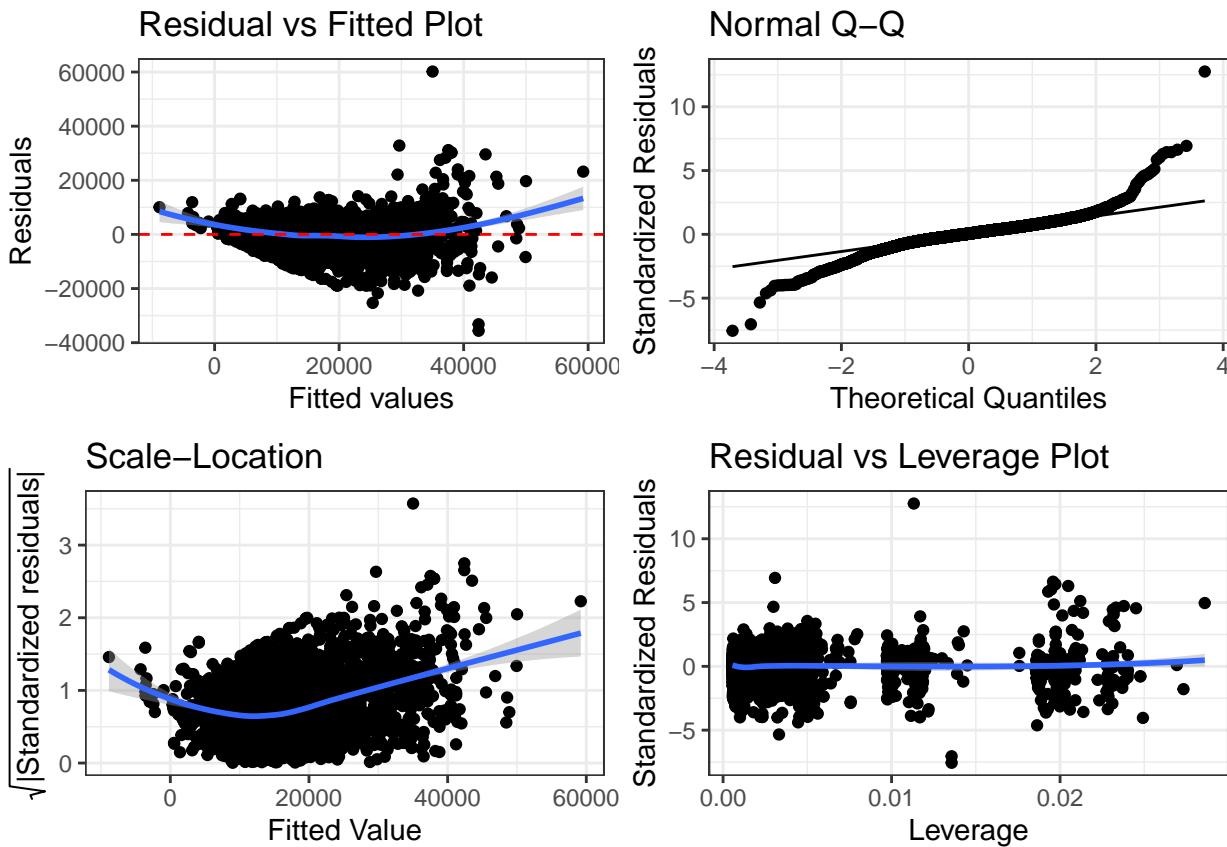
## Added-Variable Plots



```
diagPlot(key_model)
```

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

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



## Model Refinement

Based on the curved line observed in the “Residual vs Fitted” Plot in the “key\_model” and the quadratic relationship between price and mileage, we add  $\sqrt{\text{mileage}}$  to the model:  $\text{price} = \beta_0 + \beta_1 \text{mileage} + \beta_2 \text{engine power} + \beta_3 \text{model series} + \beta_4 \sqrt{\text{mileage}} + \epsilon$ . The fitted line in the “Residual vs Fitted” Plot is not as quadratic as it is in the “key\_model”. The deviated tail may be caused by potential outliers.

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

```
##
## Call:
## lm(formula = price ~ mileage + engine_power + model_series +
##   I(mileage^(1/2)), data = dat_bmw_clean)
##
## Residuals:
##   Min     1Q Median     3Q    Max 
## -37193 -2065    357   2464  57586 
## 
## Coefficients:
```

```

##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1.550e+04  8.191e+02 18.918 < 2e-16 ***
## mileage            -7.259e-03  5.992e-03 -1.211   0.2258
## engine_power        8.105e+01  2.525e+00 32.098 < 2e-16 ***
## model_series2_Series 1.602e+03  6.987e+02  2.292   0.0219 *
## model_series3_Series 2.305e+03  2.366e+02  9.746 < 2e-16 ***
## model_series4_Series 8.727e+03  5.227e+02 16.698 < 2e-16 ***
## model_series5_Series 5.062e+03  2.779e+02 18.213 < 2e-16 ***
## model_series7_Series 1.120e+04  7.227e+02 15.496 < 2e-16 ***
## model_seriesM_Power  1.552e+04  7.950e+02 19.528 < 2e-16 ***
## model_seriesOther    1.248e+04  5.702e+02 21.890 < 2e-16 ***
## model_seriesX1       1.894e+03  3.499e+02  5.411  6.57e-08 ***
## model_seriesX3       5.472e+03  3.182e+02 17.194 < 2e-16 ***
## model_seriesX5       1.342e+04  4.396e+02 30.528 < 2e-16 ***
## model_seriesX6       1.492e+04  7.962e+02 18.740 < 2e-16 ***
## I(mileage^(1/2))    -3.640e+01  4.310e+00 -8.446 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4710 on 4823 degrees of freedom
## Multiple R-squared:  0.7071, Adjusted R-squared:  0.7063
## F-statistic: 831.7 on 14 and 4823 DF,  p-value: < 2.2e-16

```

```
anova(key_model, refine_model)  #partial F-test between key_model and refine_model
```

```

## Analysis of Variance Table
##
## Model 1: price ~ mileage + engine_power + model_series
## Model 2: price ~ mileage + engine_power + model_series + I(mileage^(1/2))
##   Res.Df      RSS Df Sum of Sq    F    Pr(>F)
## 1  4824 1.0857e+11
## 2  4823 1.0699e+11  1 1582302175 71.328 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

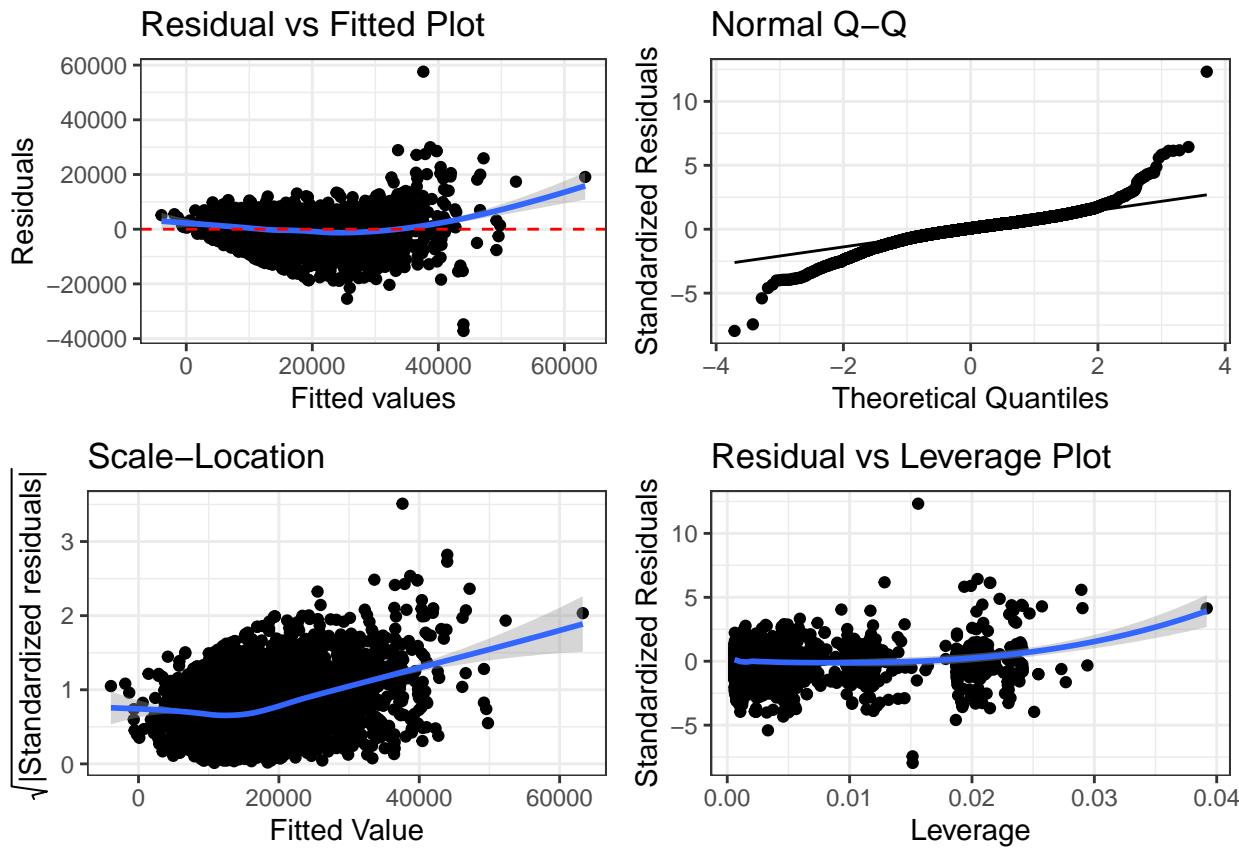
```

```
diagPlot(refine_model)
```

```

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

```



We further added the interaction term between mileage and engine\_power to see how engine\_power influence the association between price and mileage. The model was  $price = \beta_0 + \beta_1 \text{mileage} + \beta_2 \text{engine power} + \beta_3 \text{model series} + \beta_4 \sqrt{\text{mileage}} + \beta_5 \text{mileage} \times \text{engine power} + \epsilon$

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

```
##
## Call:
## lm(formula = price ~ mileage + engine_power + model_series +
##   I(mileage^(1/2)) + mileage:engine_power, data = dat_bmw_clean)
##
## Residuals:
##    Min     1Q Median     3Q    Max 
## -43422 -1962    368   2378  56422 
## 
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)             6.568e+03  9.513e+02   6.904 5.70e-12 ***
## mileage                  4.731e-02  6.635e-03   7.129 1.16e-12 ***
## engine_power              1.440e+02  4.419e+00  32.575 < 2e-16 ***
## model_series2_Series    2.289e+03  6.797e+02   3.367 0.000765 ***
## model_series3_Series    1.885e+03  2.310e+02   8.159 4.26e-16 ***
## model_series4_Series    8.160e+03  5.086e+02  16.044 < 2e-16 ***
## model_series5_Series    4.917e+03  2.700e+02  18.213 < 2e-16 ***
```

```

## model_series7_Series 1.094e+04 7.020e+02 15.582 < 2e-16 ***
## model_seriesM_Power 1.366e+04 7.796e+02 17.526 < 2e-16 ***
## model_seriesOther 1.161e+04 5.561e+02 20.875 < 2e-16 ***
## model_seriesX1 1.750e+03 3.399e+02 5.149 2.72e-07 ***
## model_seriesX3 5.241e+03 3.093e+02 16.942 < 2e-16 ***
## model_seriesX5 1.302e+04 4.275e+02 30.465 < 2e-16 ***
## model_seriesX6 1.409e+04 7.747e+02 18.182 < 2e-16 ***
## I(mileage^(1/2)) -3.108e+01 4.197e+00 -7.405 1.54e-13 ***
## mileage:engine_power -4.723e-04 2.761e-05 -17.109 < 2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4574 on 4822 degrees of freedom
## Multiple R-squared: 0.7239, Adjusted R-squared: 0.723
## F-statistic: 842.8 on 15 and 4822 DF, p-value: < 2.2e-16

```

```
anova(refine_model, interact_model)
```

```

## Analysis of Variance Table
##
## Model 1: price ~ mileage + engine_power + model_series + I(mileage^(1/2))
## Model 2: price ~ mileage + engine_power + model_series + I(mileage^(1/2)) +
##   mileage:engine_power
##   Res.Df      RSS Df  Sum of Sq      F    Pr(>F)
## 1 4823 1.0699e+11
## 2 4822 1.0087e+11  1 6123355871 292.73 < 2.2e-16 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

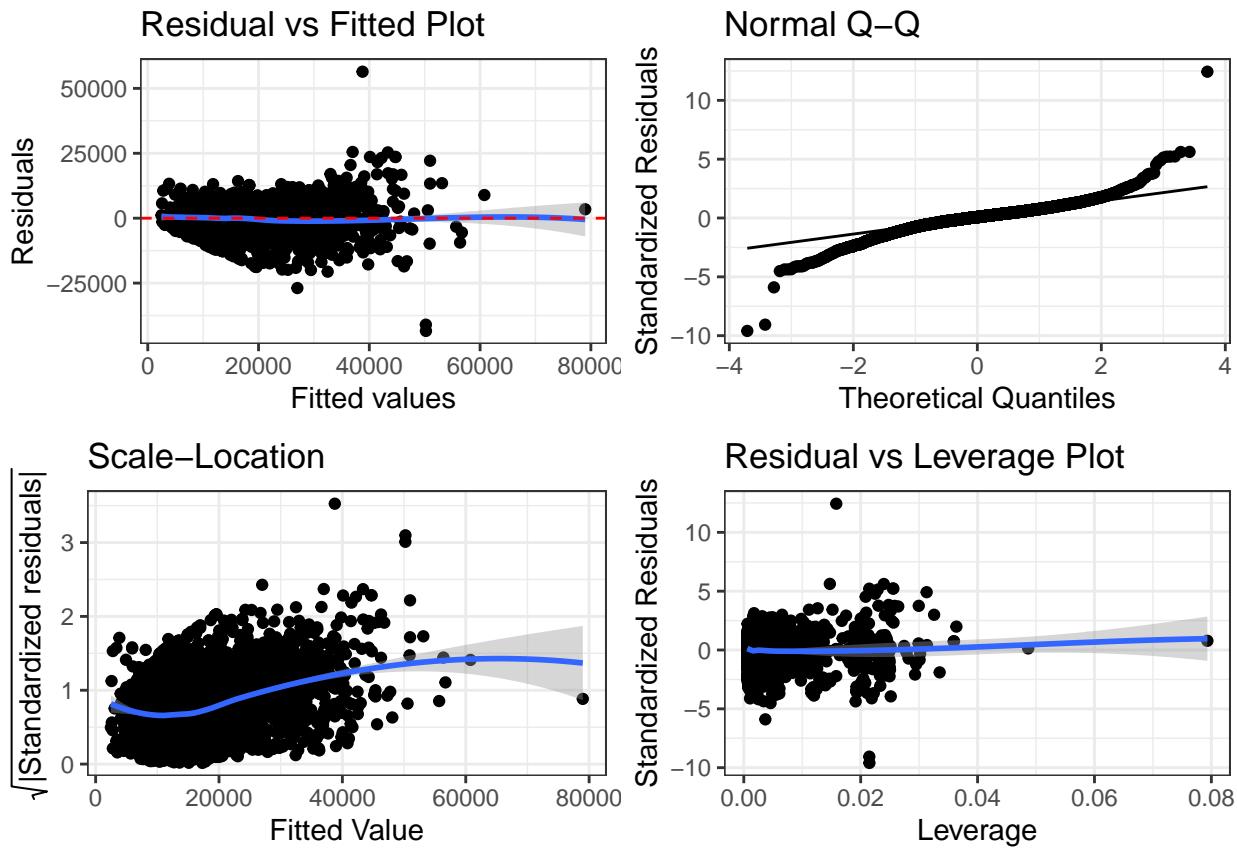
```

```
diagPlot(interact_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))
```

# 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

```

```

# 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 "-0.12278957659823"  "-0.145500785077517"  "116"
## age "-0.143433541715236"  "-0.162697652058148"  "118"
## age "-0.149038443543364"  "-0.171057626029924"  "316"
## age "-0.136161010233223"  "-0.149777693349068"  "318"
## age "-0.16563587460071"   "-0.178490627890781"  "320"
## age "-0.183404655126876"  "-0.191612478867974"  "520"
## age "-0.206231690635819"  "-0.224201035913407"  "525"
## age "-0.185604933819817"  "-0.198075674413492"  "530"
## age "-0.0182544527786139"  "-0.0676029110385667"  "X1"
## age "-0.11571932710388"   "-0.13260525273682"   "X3"
## age "-0.171213304919039"  "-0.177936709330697"  "X5"

```

```

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)

```

```

##      age:is_X1 + age:is_X3 + age:is_116, data = dat_age)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -35980 -1533     110    1807  53514
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)           1.327e+04  8.165e+02 16.258 < 2e-16 ***
## mileage              1.126e-01  5.654e-03 19.918 < 2e-16 ***
## engine_power          1.445e+02  3.645e+00 39.643 < 2e-16 ***
## model_series2_Series 4.833e+02  5.894e+02  0.820  0.41221
## model_series3_Series 1.496e+03  2.724e+02  5.492 4.19e-08 ***
## model_series4_Series 7.114e+03  4.545e+02 15.654 < 2e-16 ***
## model_series5_Series 4.903e+03  2.852e+02 17.192 < 2e-16 ***
## model_series7_Series 1.240e+04  6.010e+02 20.631 < 2e-16 ***
## model_seriesM_Power   1.415e+04  6.591e+02 21.473 < 2e-16 ***
## model_series0Other    1.192e+04  4.873e+02 24.472 < 2e-16 ***
## model_seriesX1        -1.629e+03  1.106e+03 -1.473  0.14094
## model_seriesX3        5.742e+03  5.442e+02 10.552 < 2e-16 ***
## model_seriesX5        1.393e+04  3.895e+02 35.759 < 2e-16 ***
## model_seriesX6        1.539e+04  6.577e+02 23.396 < 2e-16 ***
## I(mileage^(1/2))     -5.259e+01  3.501e+00 -15.020 < 2e-16 ***
## age                   -1.239e+03  2.712e+01 -45.710 < 2e-16 ***
## mileage:engine_power -5.414e-04  2.286e-05 -23.680 < 2e-16 ***
## age:is_318            7.269e+01  3.008e+01  2.417  0.01569 *
## age:is_X1             6.289e+02  2.028e+02  3.102  0.00193 **
## age:is_X3             -4.629e+01  8.148e+01 -0.568  0.56998
## age:is_116            1.145e+02  5.594e+01  2.047  0.04073 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3761 on 4817 degrees of freedom
## Multiple R-squared:  0.8134, Adjusted R-squared:  0.8127
## F-statistic: 1050 on 20 and 4817 DF,  p-value: < 2.2e-16

```

```
vif(age_model)
```

```

## there are higher-order terms (interactions) in this model
## consider setting type = 'predictor'; see ?vif

```

```

##                               GVIF Df GVIF^(1/(2*Df))
## mileage                  37.904179  1      6.156637
## engine_power              6.888626  1      2.624619
## model_series              814.487760 11     1.356166
## I(mileage^(1/2))         28.659254  1      5.353434
## age                      1.602662  1      1.265963
## mileage:engine_power     15.477916  1      3.934198
## age:is_318                1.333149  1      1.154621

```

```

## age:is_X1           21.331302  1      4.618582
## age:is_X3           6.746347  1      2.597373
## age:is_116          2.314989  1      1.521509

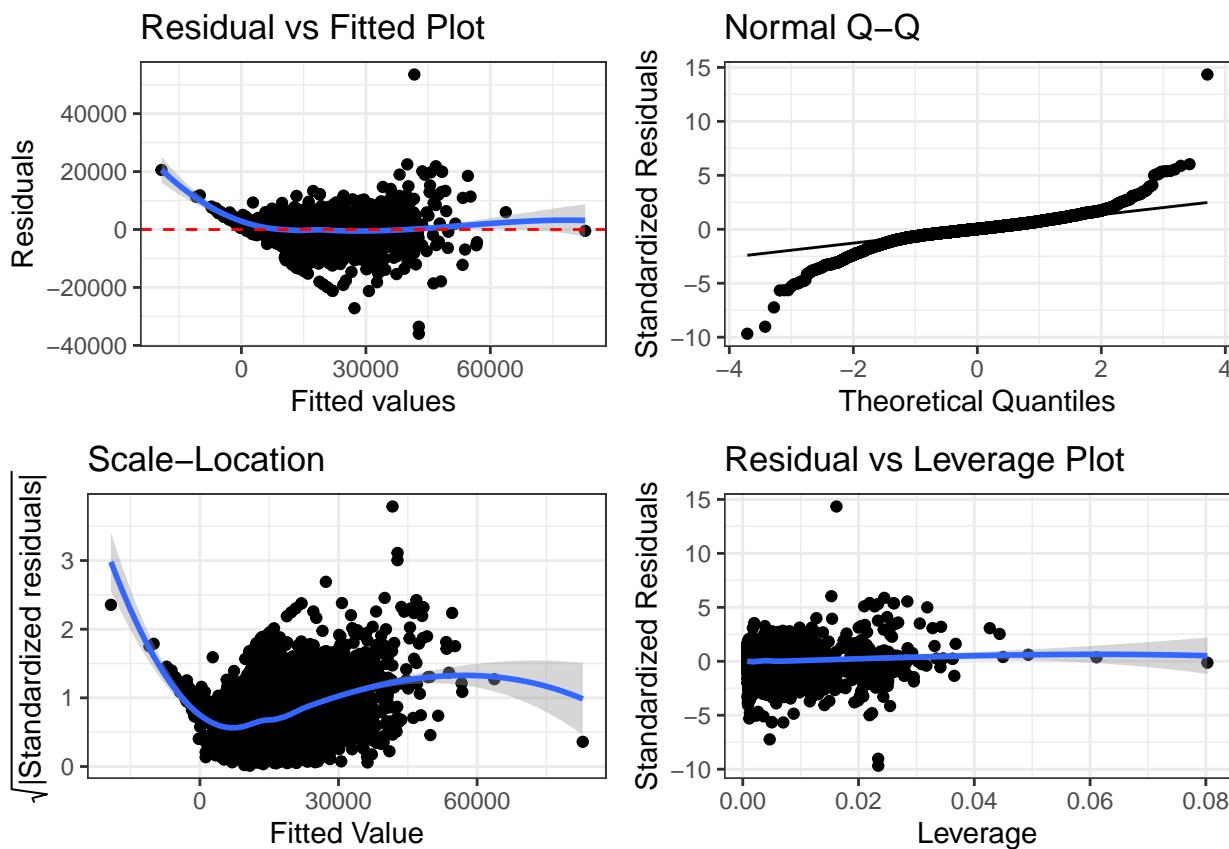
```

```
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:  $price = \beta_0 + \beta_1 mileage + \beta_2 engine\ power + \beta_3 model\ series + \beta_4 \sqrt{mileage} + \beta_5 mileage \times engine\ power + \beta_6 age + \epsilon$ .

```

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)

```

```

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

## 
## Call:
## lm(formula = price ~ mileage + engine_power + model_series +
##     I(mileage^(1/2)) + mileage:engine_power + age, 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 ***
## engine_power              1.429e+02  3.656e+00 39.100 < 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 ***
## I(mileage^(1/2))       -5.126e+01  3.498e+00 -14.653 < 2e-16 ***
## age                     -1.206e+03  2.556e+01 -47.171 < 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

```

```
anova(interact_model, final_model)
```

```

## Analysis of Variance Table
## 
## Model 1: price ~ mileage + engine_power + model_series + I(mileage^(1/2)) +
##     mileage:engine_power
## Model 2: price ~ mileage + engine_power + model_series + I(mileage^(1/2)) +
##     mileage:engine_power + age

```

```

##   Res.Df      RSS Df Sum of Sq    F    Pr(>F)
## 1 4822 1.0087e+11
## 2 4821 6.9014e+10  1 3.1853e+10 2225.1 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

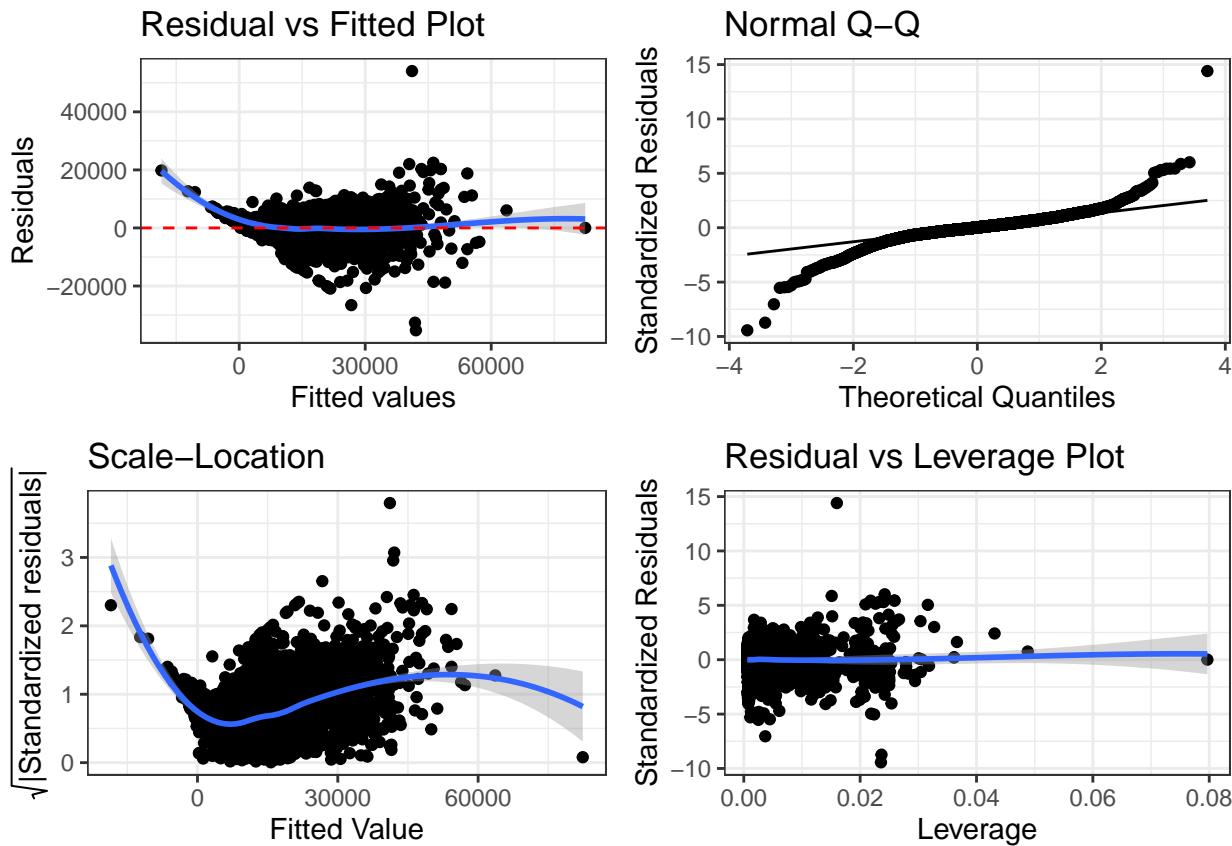
```

```
diagPlot(final_model)
```

```

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

```



## Modelling and Analysis for the Training Data Set

### Split Training and Testing Data

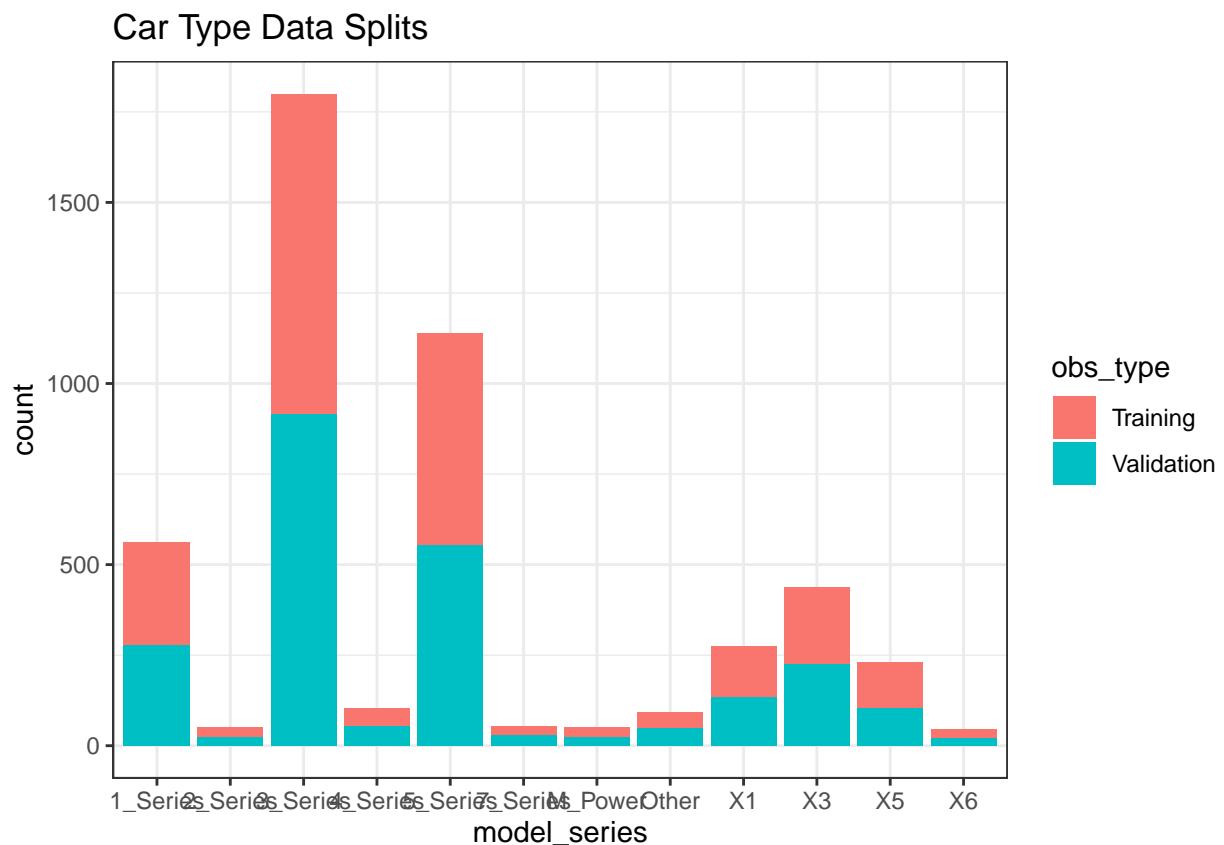
```

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, ]

```

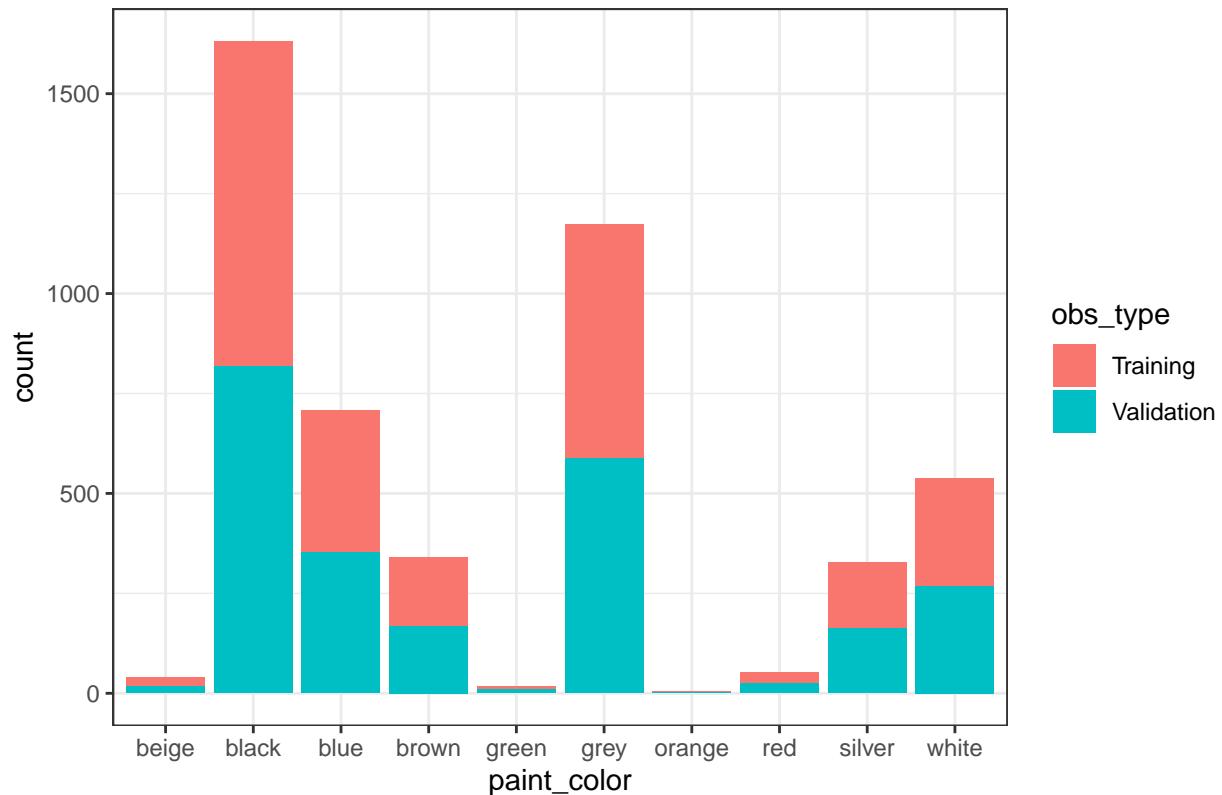
## Data Overview of the Training and Testing Data

```
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()
```



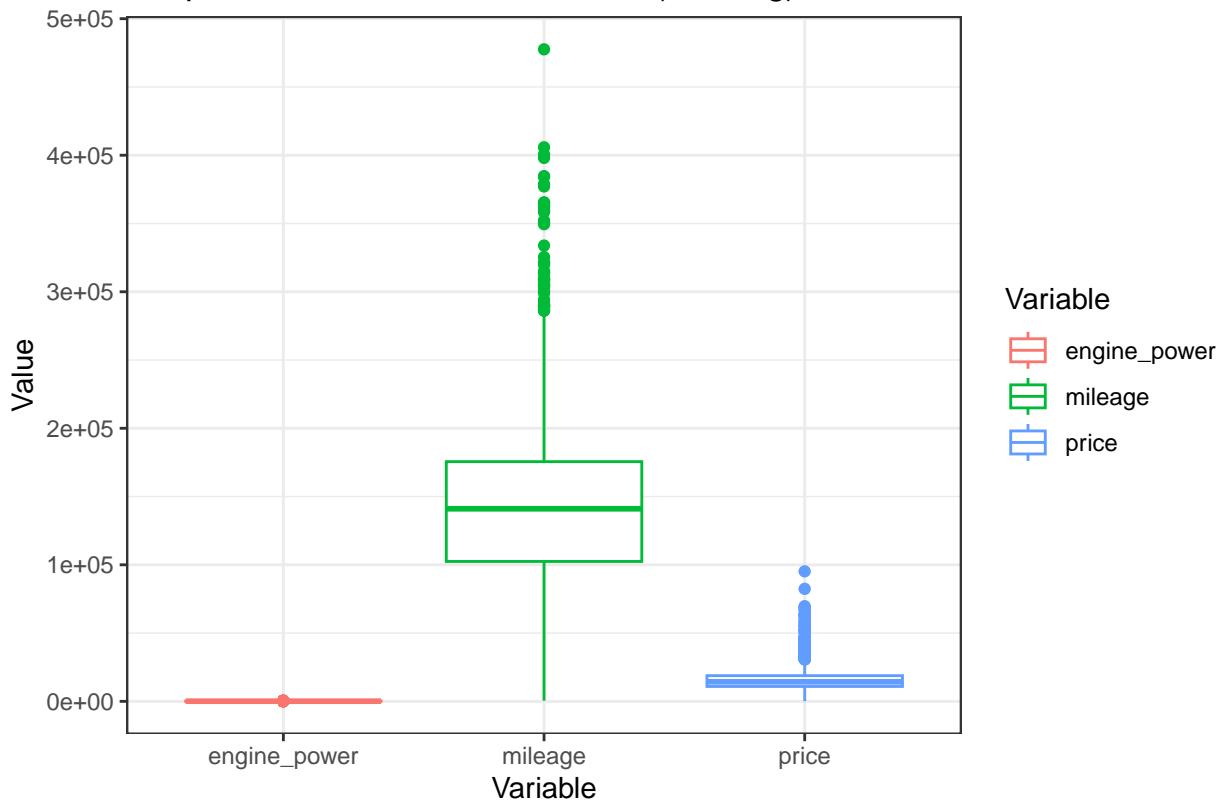
```
dat_bmw_clean |>
  ggplot(aes(x = paint_color, fill = obs_type)) + geom_bar() +
  labs(title = "Car Type Data Splits") + theme(axis.text.x = element_text(angle =
  -30)) +
  theme_bw()
```

## Car Type Data Splits



```
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()
```

## Boxplots for Continuous Variables (Training)



## 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_series0Other 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

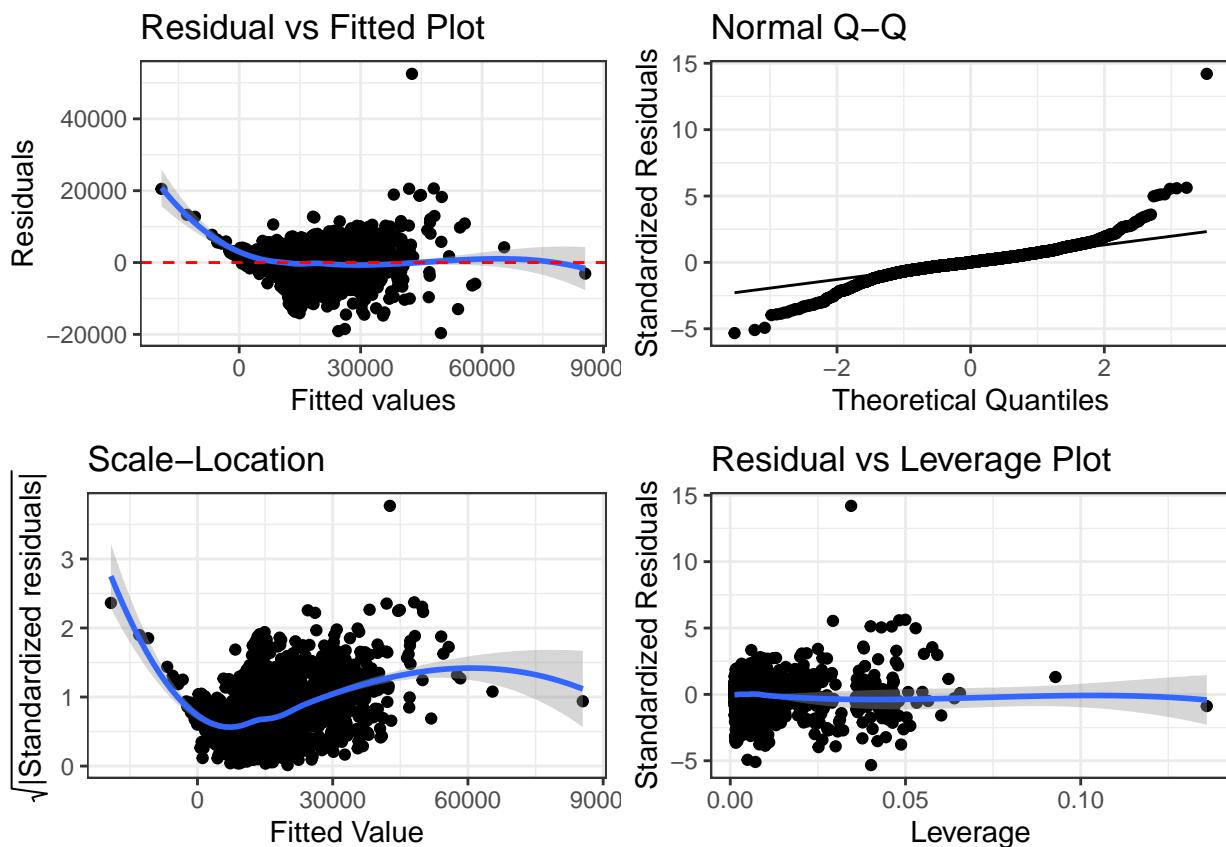
```

```
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.

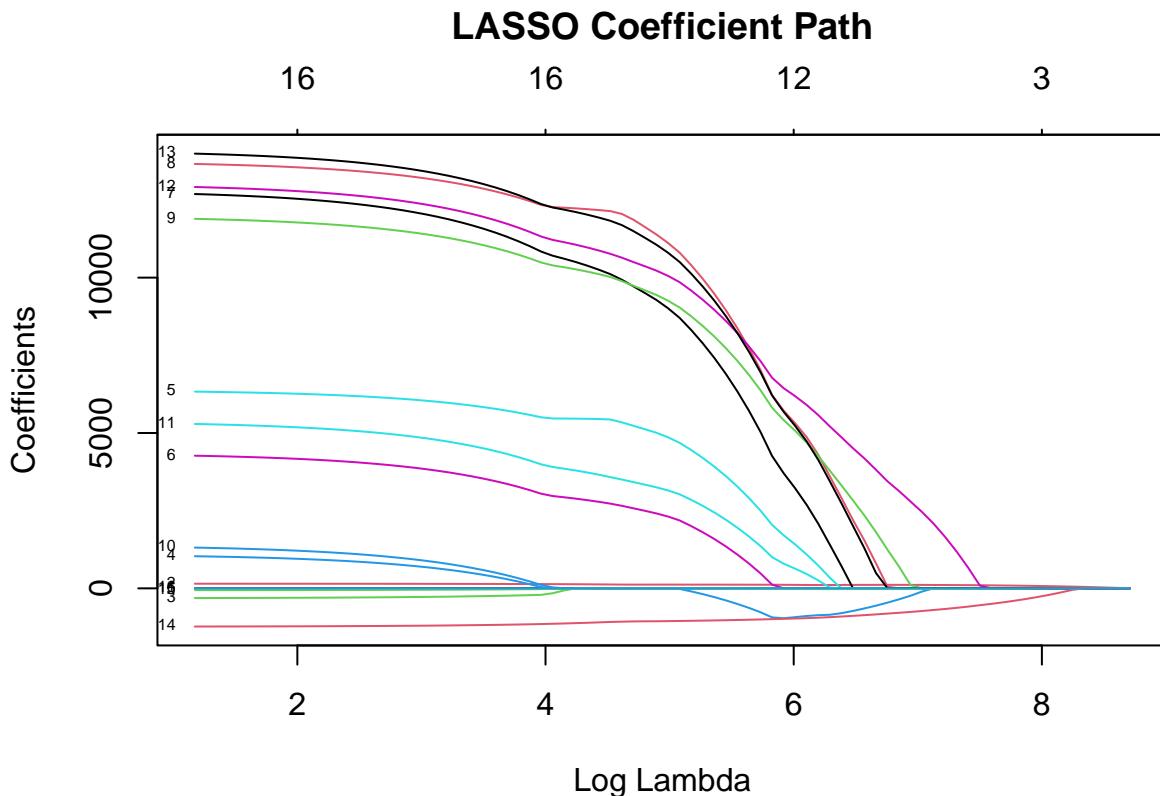
```
# Create a model matrix
X <- model.matrix(price ~ mileage + engine_power + model_series +
  age + I(mileage^(1/2)) + mileage:engine_power, data = dat_bmw_train)[,
  -1]

# Create the response variable
Y <- dat_bmw_train$price

# set seed for reproducibility
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.

```

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.,  $\beta_{\text{mileage} \times \text{engine power}} = -5.87 \times 10^{-4}$ ). 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_series0Other   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]

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

summary(dat_bmw_test$price)

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

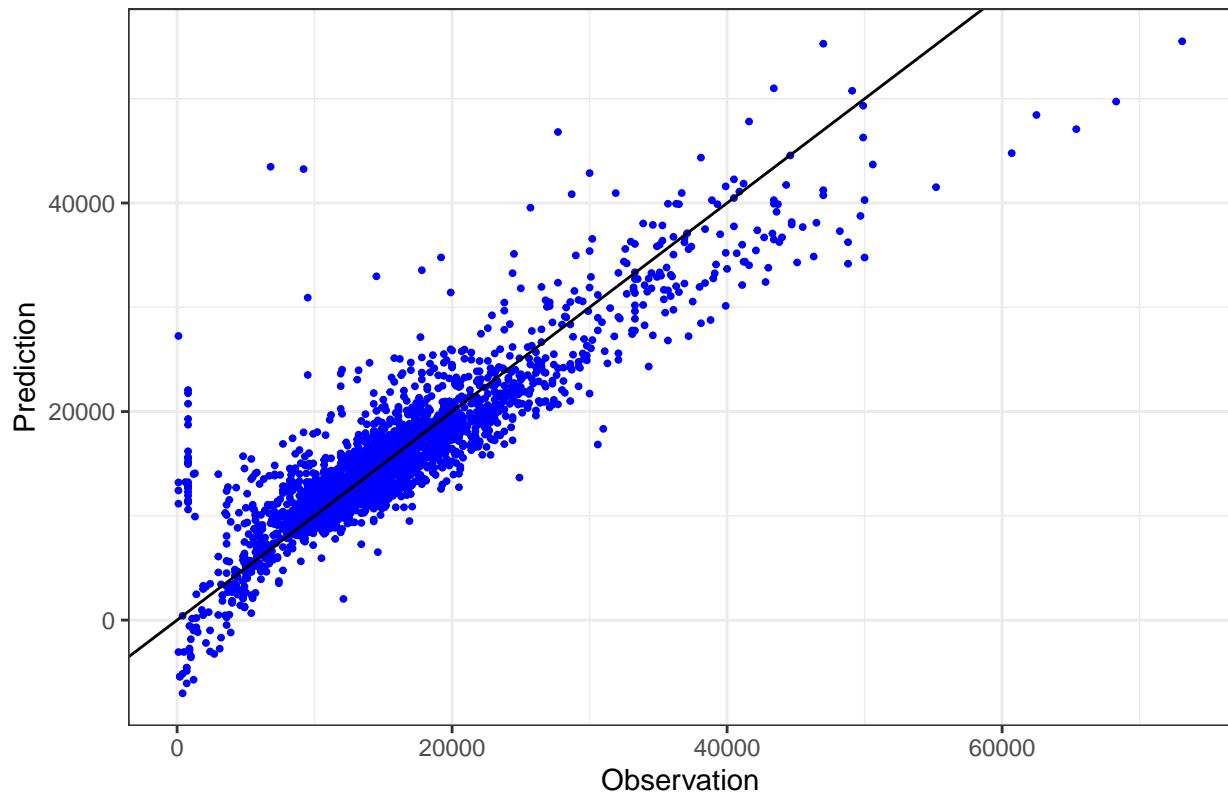
summary(dat_bmw_clean$price)

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

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 = "Comparison between Observed
  and Predicted Price") +
  xlab("Observation") + ylab("Prediction") + geom_abline(slope = 1,
  intercept = 0) + theme_bw()
```

## Comparesion between Observed and Predicted Price



```
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()
```

## Comparision between Observed and Predicted Price

