Group P: Project Analysis Component

Sydney Bluestein, Parker Smith, Tate Eppinger, and Juan Perez

2024-04-03

```
Cars <- janitor::clean_names(Sport_car_price)
Cars <- unique(Cars)
Cars <- Cars %>%
  filter(!grepl("Electric", engine_size_l)) %>%
  mutate(across(engine_size_l:x0_60_mph_time_seconds, as.numeric))
Cars2<-Cars[-c(94, 150, 154, 409),] #deletes middle eastern cars and super old car
Cars3<-filter(Cars2, price_in_usd<500000) #removed data from IQR test below
Cars4<-na.omit(Cars3)
## MAR: Recategorized into "Asian", "Domestic", and "European"
SportsCars <- Cars4 %>% mutate(car_make = case_when(
    car_make %in% c("Acura", "Kia"  , "Lexus"  , "Mazda"  , "Nissan" , "Subaru" , "Toyota") ~ "Asian",
    car_make %in% c("Ford" , "Chevrolet" , "Dodge") ~ "Domestic",
    car_make %in% c("Alfa Romeo" , "Alpine" , "Ariel" , "Aston Martin" , "Audi" , "Bentley", "BMW", "Bu
```

Does "what's under the hood" matter when predicting sports car prices in America? vroom vroom Group P: Sydney Bluestein, Tate Eppinger, Juan Perez, and Parker Smith

Project Aim: We want to explore which factors contribute to sports car prices in America. We have chosen numerous explanatory variables that may influence sports car prices: car origin, car year, 0-60 speed, torque, horsepower, and engine size.

Research Question: Do any of the factors above (or a combination of them) predict sports car prices accurately?

Variables: Car origin (Asian, Domestic, or European) is our categorical variable. Year is a quantitative variable that represents the year the car was made. 0-60 speed is a quantitative variable that represents the time in seconds it takes for the car to go from 0 miles per hour to 60 miles per hour. Torque is a quantitative variable measured in pounds-feet that describes the power of the car's drivetrain. Horsepower is another quantitative variable measured in foot-pounds/minute representing engine power. Lastly, engine size is a quantitative variable in liters that measures how much gas the car can hold.

Step 1 – Univariate Analysis: We included a brief univariate analysis in our project proposal. We included density plots of horsepower, torque, engine size, and 0-60. It would be useful to obtain descriptive stats for these quantitative variables using the favstats method. Looking into the quantitative variables' shape, center, and spread will help orient us toward recognizing typical and unusual values for variables measured in unfamiliar units. Car year is characterized by a box plot. We removed a few cars produced before 2014 since we are attempting to predict sports car prices from the last decade. We also included a bar graph for our categorical variable, car origin. We notice that European cars are most commonly represented in this data set. However, there are still enough Asian and domestic sports cars to make this variable worthwhile. We will use the tally function to create a table illustrating the exact distribution of cars originating from Europe, Domestic, and Asia.

```
m1<-gf_dens(~ horsepower, data = SportsCars, color = "blue")
m2<-gf_dens(~ torque_lb_ft, data = SportsCars, color = "deepskyblue")
m3<-gf_dens(~ engine_size_l, data = SportsCars, color = "pink2")
```

```
m4<-gf_dens(~ x0_60_mph_time_seconds, data = SportsCars,  xlab="0 to 60mph time in seconds", color = "pu
```

```
m5<-gf_bar (~car_make, data= SportsCars)
m6<-gf_boxplot(~year, data=SportsCars)
kable(favstats(~year, data = SportsCars), caption = "Year")  #IQR test 1.5, 1.5(IQR=1)=1.5, data range
```

Table 1: Year

min	Q1	median	Q3	max	mean	sd	n	missing
2014	2021	2021	2022	2023	2021.233	1.015807	592	0

```
kable(favstats(~horsepower, data = SportsCars), caption = "Horsepower")
```

Table 2: Horsepower

min	Q1	median	Q3	max	mean	sd	n	missing
181	416	505	626	1200	528.7703	135.2675	592	0

```
kable(favstats(~torque_lb_ft, data = SportsCars), caption = "Torque (lb-ft)")
```

Table 3: Torque (lb-ft)

min	Q1	median	Q3	max	mean	sd	n	missing
151	384	468	590	1300	473.9257	130.5958	592	0

```
kable(favstats(~engine_size_l, data = SportsCars), caption = "Engine Size")
```

Table 4: Engine Size

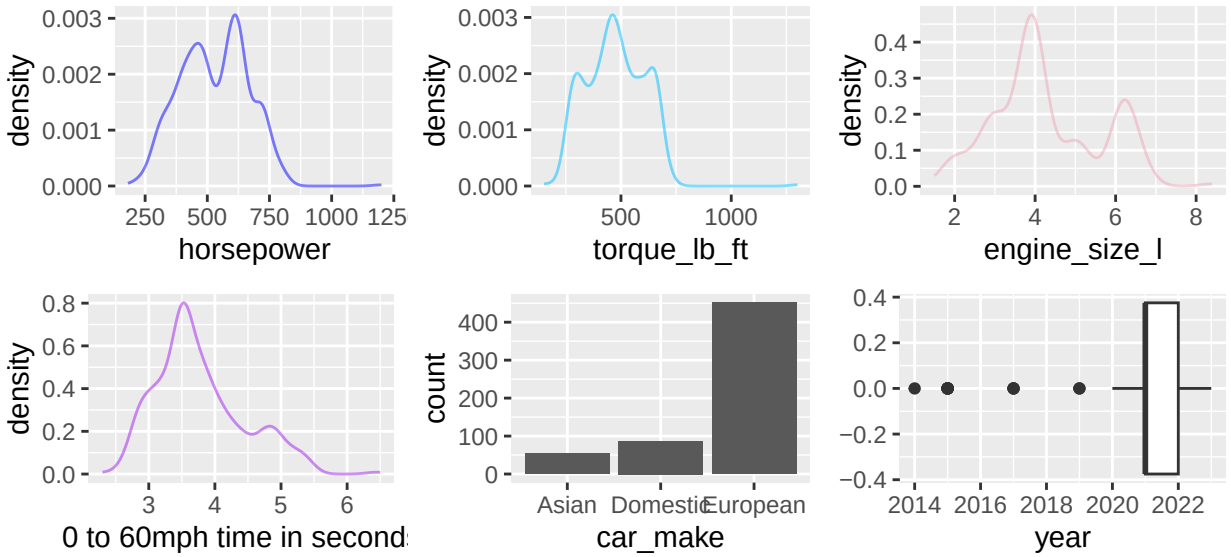
min	Q1	median	Q3	max	mean	sd	n	missing
1.5	3.5	4	5.2	8.4	4.27196	1.331942	592	0

```
kable(favstats(~x0_60_mph_time_seconds, data = SportsCars), caption = "0 to 60 mph time")
```

Table 5: 0 to 60 mph time

min	Q1	median	Q3	max	mean	sd	n	missing
2.3	3.3	3.6	4.1	6.5	3.773311	0.676082	592	0

```
grid.arrange(m1, m2, m3, m4, m5, m6, ncol=3, heights=c(2, 2, 2))
```



Horsepower - The plot shows a bimodal distribution with peaks around 450 horsepower and 600 horsepower, indicating two prevalent groups of sports cars with distinct power outputs in this dataset. The median of 505 horsepower indicates that half of the cars have horsepower above this value, and half below, showcasing the central tendency of the distribution. The maximum horsepower is 797, highlighting the presence of very high-powered vehicles within the dataset. The mean horsepower is approximately 527.6345, which is close to the median, reinforcing the bimodal distribution seen in the density plot as it suggests a symmetry in the distribution of values. Additionally, the standard deviation of about 132.5269 horsepower indicates a moderate spread around the mean.

Torque - The density plot for the torque (measured in pound-feet) of sports cars in the dataset shows a bimodal distribution with peaks roughly around 300 and 500 lb-ft, suggesting two prevalent groups of sports cars based on their torque characteristics. The minimum torque is 151 lb-ft, indicating the lowest torque present in a sports car in this dataset. The median of 468 lb-ft represents the middle value, where half the cars have more torque and the other half have less. The maximum value is 738 lb-ft, showcasing some sports cars with very high torque outputs. The mean torque is around 472.5279 lb-ft, which is very close to the median, indicating a relatively symmetrical distribution around the central value. The standard deviation of approximately 126.1966 lb-ft indicates a moderate variability in the torque figures across the dataset.

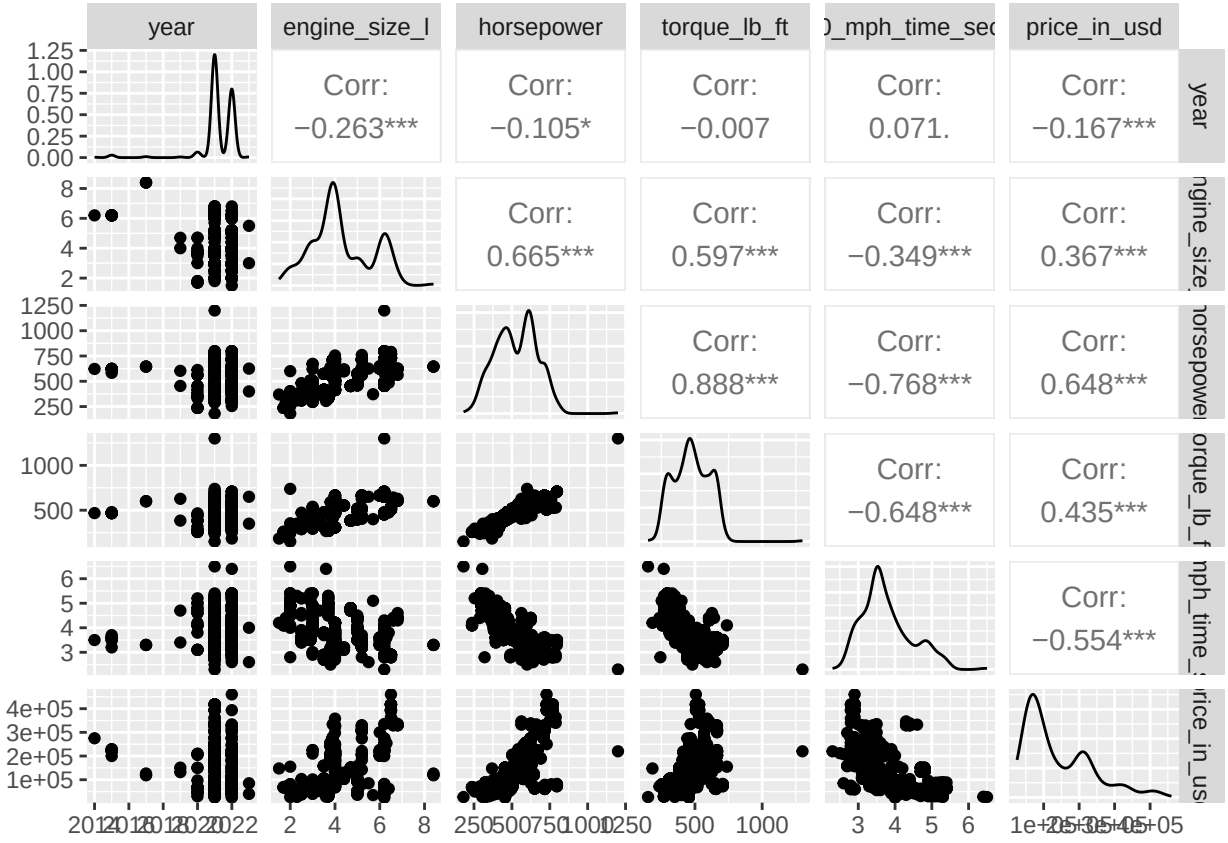
Engine Size - The density plot for engine size (in liters) reveals a bimodal distribution with two prominent peaks: one just below 4 liters and another around 5 liters. The minimum size of 1.5 liters and a maximum of 8.4 liters, suggesting a wide range of engine sizes among the sports cars. The median engine size is 4 liters, which lies between the two peaks of the density plot, and the mean engine size is approximately 4.27 liters, indicating a slight skew to the right. The standard deviation is around 1.33 liters, which points to a relatively wide spread of engine sizes in the dataset. The mean being greater than the median suggests a right skew in the data, indicating that there are sports cars with larger engine sizes that are pulling the average to the right. This skew is likely not very strong but does suggest the presence of more sports cars with higher engine sizes than those with smaller ones.

0 to 60 MPH Time - The density plot for the 0 to 60 mph acceleration timesshows that the most common acceleration time is around 3.5 seconds, which is where the peak of the density occurs. The range of acceleration times is from 2.5 seconds to 6.5 seconds, with a median time of 3.6 seconds. The mean acceleration time is slightly higher at 3.775804 seconds, suggesting a slight right skew in the distribution. This is further supported by the mean being greater than the median, implying that there are more sports cars with slightly slower acceleration times pulling the average to the right. The standard deviation of about 0.673926 seconds shows a relatively tight clustering of times around the mean.

Step 2 – Bivariate Analysis: We will begin with simple linear regression to gauge which explanatory variables help predict sports car prices. It will be time-efficient to begin with GGpairs as a summary of the relationships

between our explanatory variables and sports car prices. Upon the first glimpse, the GGpairs plot demonstrates that horsepower (r=0.648) and 0-60 (r=-0.554) may be helpful predictors for sports car prices. We plan to explore this relationship further by creating simple linear regression models. To check for linearity, we can look either at the SLR plot or the residuals vs. fitted plot to ensure the relationship between our predictor variable and sports car price is best described as linear. Independence is not a condition we can assume from a plot. (We have very little information regarding the origin of this dataset, so we will have to proceed with caution). Looking at whether the data points fit well on the expected line QQ plot line will allow us to decide whether the normality condition is satisfied. We will confirm homoscedasticity for the equal variance condition by looking for a seemingly random distribution of points on the residuals vs. fitted plot. If we encounter any issues with conditions, we will look into removing potential outliers or transforming the data if necessary.

```
ggpairs(SportsCars, columns = c(3:8))
```



Step 3 – Model Building and Variable Selection: Once we have looked into SLR, we will turn to multiple linear regression. Looking at the GGpairs plot, we notice many potential pairs of predictors run the risk of being collinear. (torque and horsepower, 0-60 and horsepower, and horsepower and engine size). We can run a VIF test if we choose to include variables that may be collinear to confirm we have created a stable model. We may use an automated variable selection method (maybe stepwise, so we get the best predictors at every stage) to delineate which combination of explanatory variables will help predict sports car prices best. We will then weigh the models' parsimony with adjusted R-squared values and Mallow's Cp to determine the best model(s). If we have a potential outlier, we can use a Cook's distance plot to determine if the point is unusual enough to remove from our model.

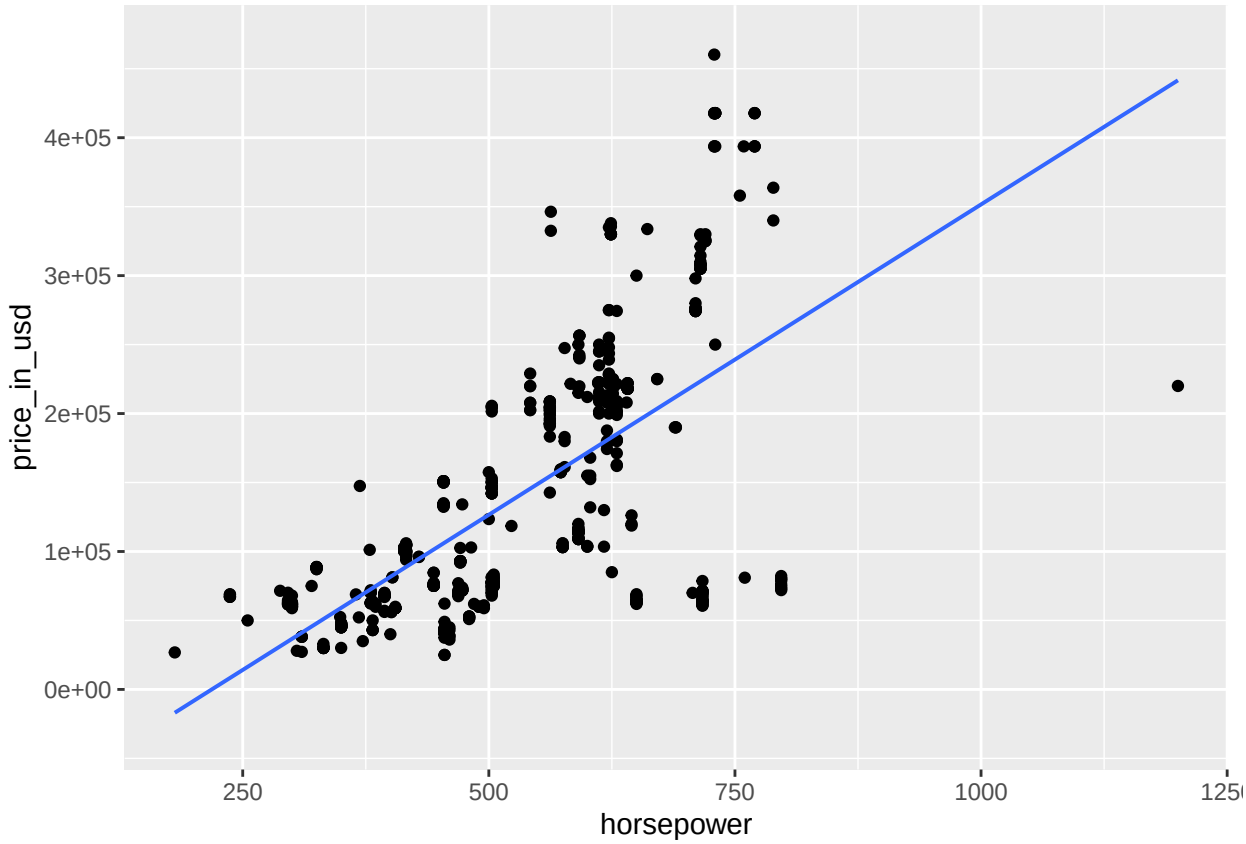
Regarding our categorical variable, we would like to run an ANOVA test to determine if the mean sports car price differs based on its region of origin. Suppose the ANOVA test is associated with a significant F statistic. We will then run a Tukey test (or a similar multiple comparison test) to determine which region(s) of origin have significantly different sports car prices. The results from this multiple comparison test will

help us determine how we need to adjust our model, i.e., for what region of origin do we need to account for higher sports car prices relative to the other regions? Perhaps we would want to look into some interaction terms as well.

SLR: Horsepower and 0-60 appeared to be our strongest predictors for car price. However the scatterplot of horsepower vs. car price proved to be pretty curvy. I attempted to transform the data by predicting log (car price) rather than car price, log(horsepower) instead of horsepower, etc. However, conditions still were not met. Similarly for 0-60, conditions were not met for SLR. I removed an outlier, representing a car with extremely high torque and horsepower. I believed this point was pulling the regression line down. After removing this point, I then transformed the data by having -1/sqrt(0-60) predict price. However, conditions for SLR still failed to be met. I then wanted to check if using horsepower and 0-60 together would provide a better fitting model than the predictors on their own. Most of the points in the fitted vs. residuals plot are shifted to the right, demonstrating that there is not perfectly equal variance. Also the QQ plot appears curvy, however it looks less sigmoidal than any of the SLR model QQ plots. Though conditions are still not met perfectly, this model fits better than either of the SLR models. We will consider this bivariate model with caution.

```
gf_point(price_in_usd ~horsepower, data=SportsCars) %>% gf_lm()
```

```
## Warning: Using the `size` aesthetic with geom_line was deprecated in ggplot2 3.4.0.
## i Please use the `linewidth` aesthetic instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



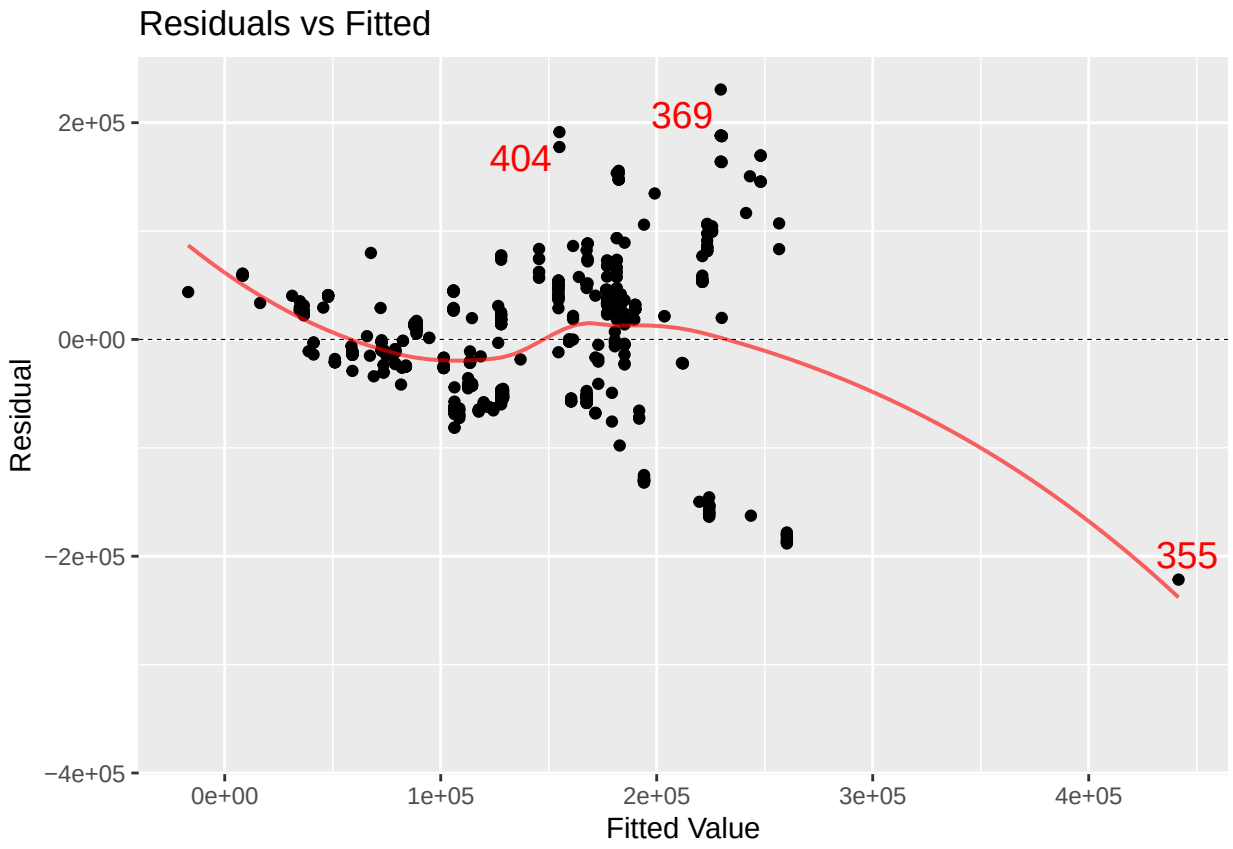

```
horsepowerSLR<- lm(price_in_usd ~horsepower, data=SportsCars)
msummary(horsepowerSLR)
```

```
##               Estimate Std. Error t value Pr(>|t|)
```

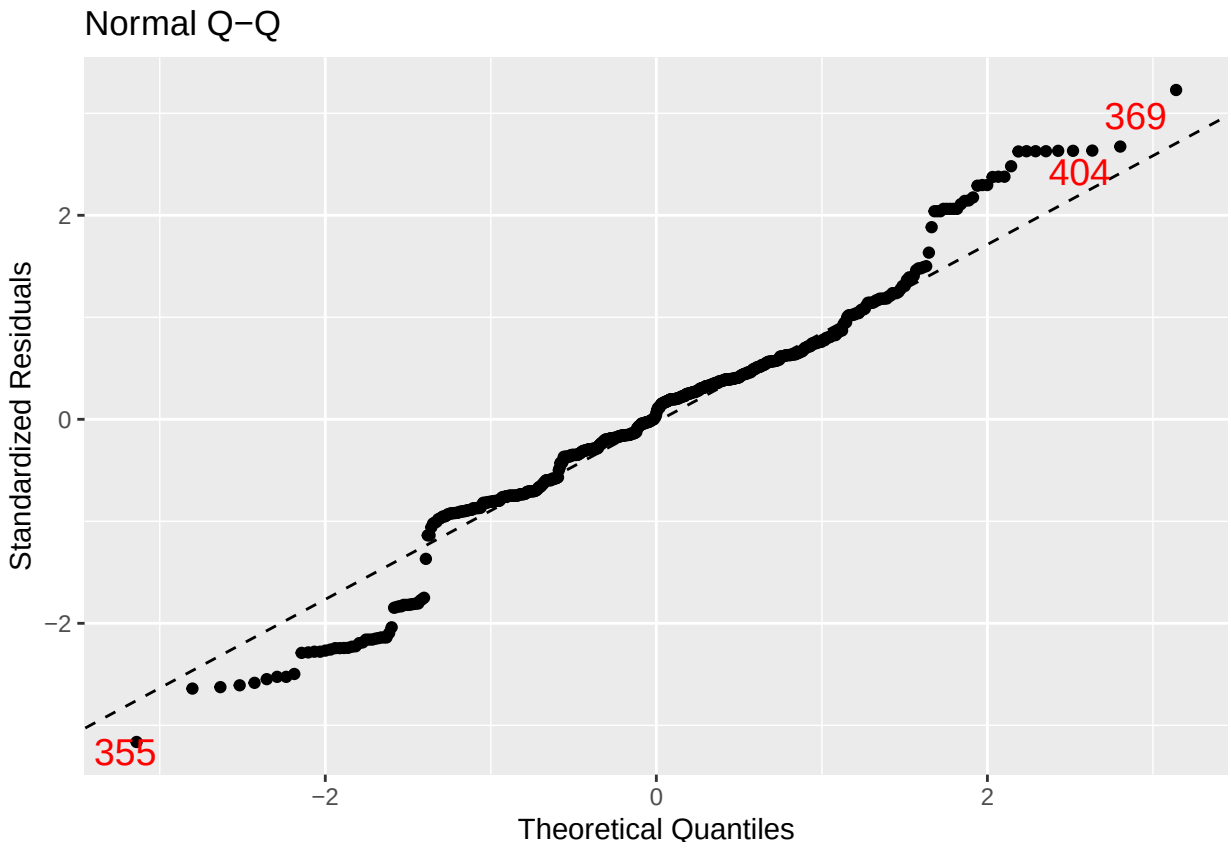
```
## (Intercept) -98357.07   11886.18  -8.275 8.59e-16 ***
## horsepower     449.96      21.78  20.660  < 2e-16 ***
##
## Residual standard error: 71620 on 590 degrees of freedom
## Multiple R-squared:  0.4198, Adjusted R-squared:  0.4188
## F-statistic: 426.9 on 1 and 590 DF,  p-value: < 2.2e-16
```

```
mplot(horsepowerSLR, which=1)
```

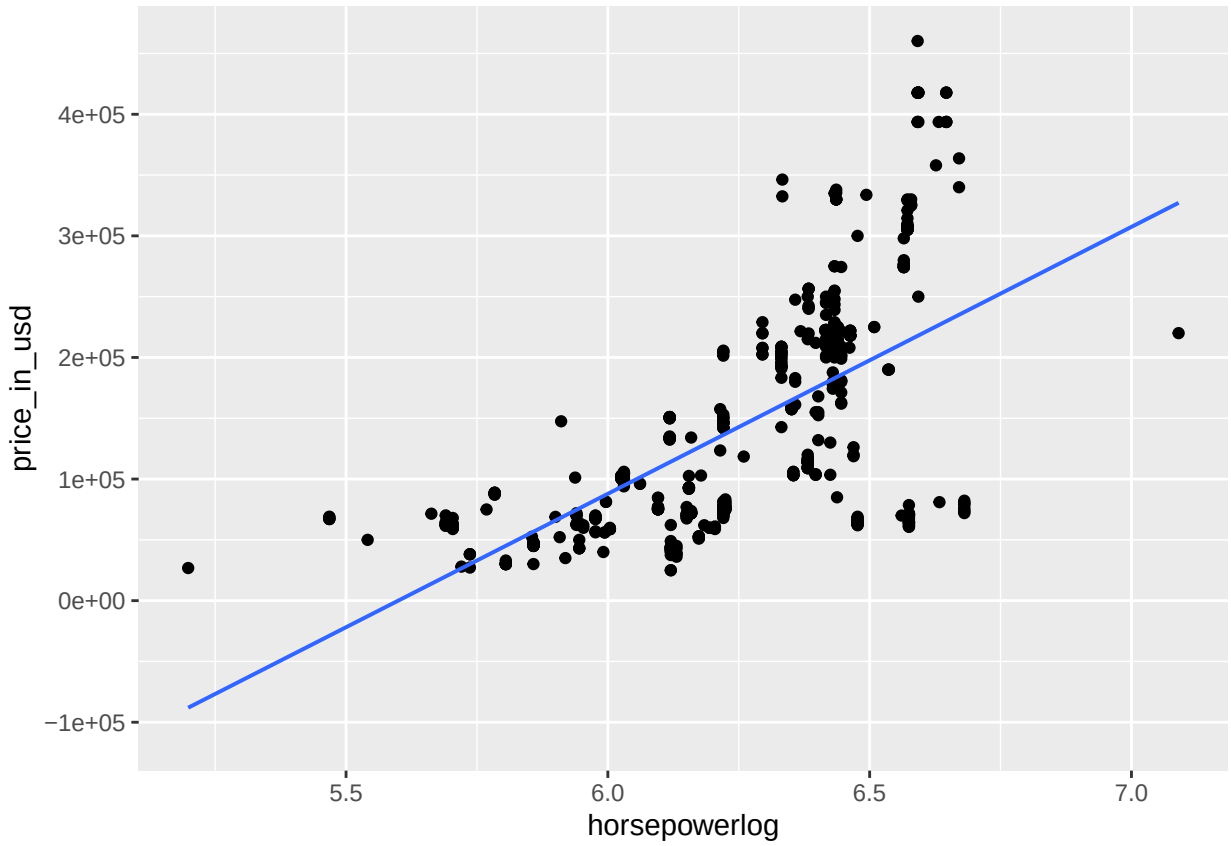
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
mplot(horsepowerSLR, which=2)
```




```
SportsCars <- mutate(SportsCars, horsepowerlog = log(horsepower))
gf_point(price_in_usd ~horsepowerlog, data=SportsCars) %>% gf_lm()
```

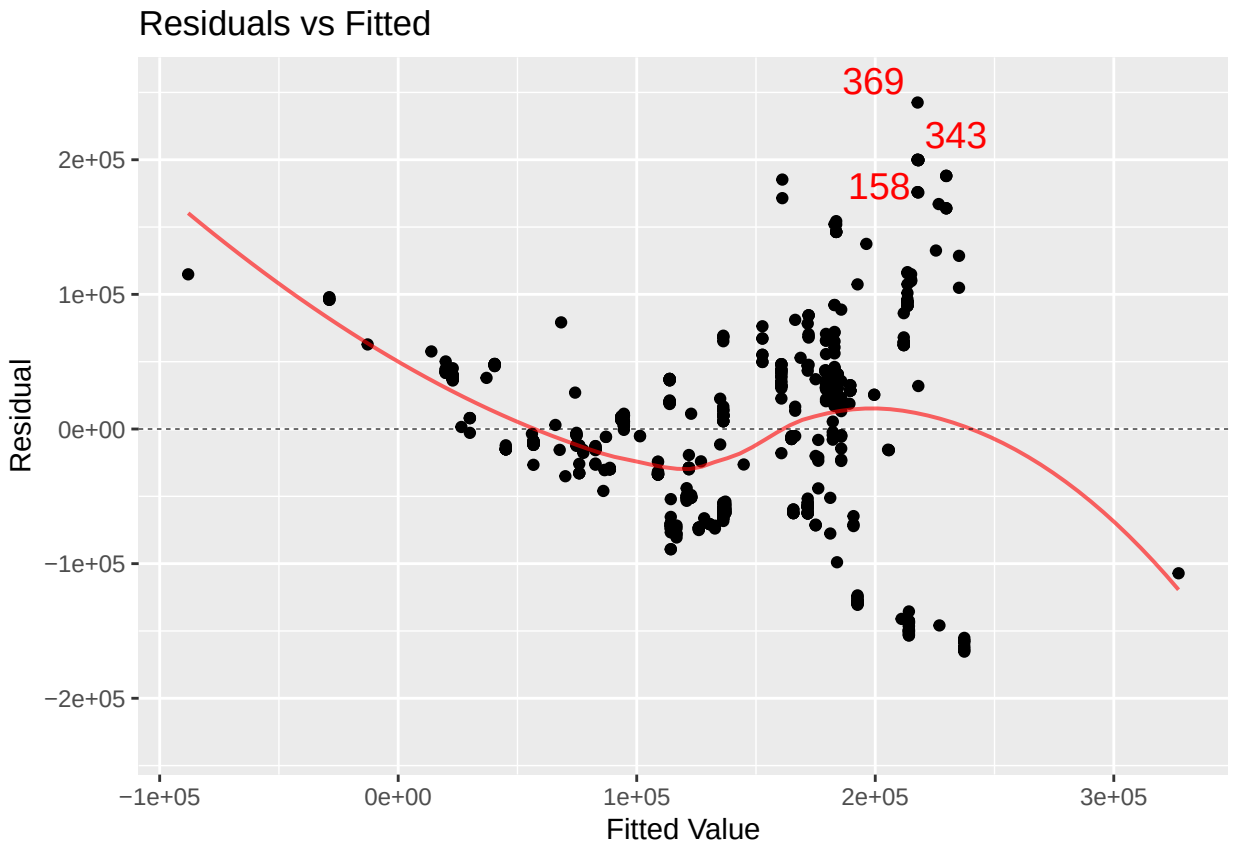



```
horsepowerSLR3<- lm(price_in_usd ~horsepowerlog, data=SportsCars)
msummary(horsepowerSLR3)
```

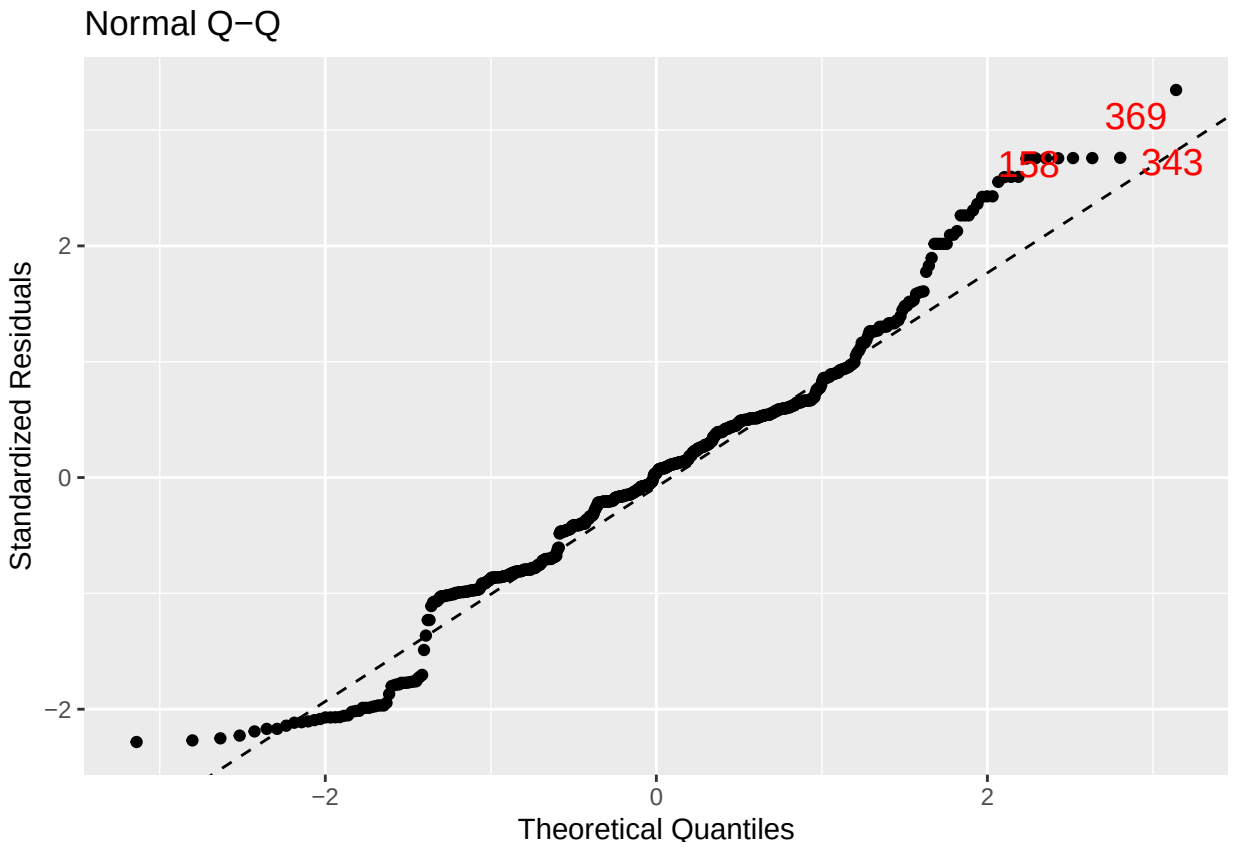
```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -1229145      68642  -17.91   <2e-16 ***
## horsepowerlog   219504      10998   19.96   <2e-16 ***
##
## Residual standard error: 72640 on 590 degrees of freedom
## Multiple R-squared:  0.403,  Adjusted R-squared:  0.402
## F-statistic: 398.3 on 1 and 590 DF,  p-value: < 2.2e-16
```

```
mplot(horsepowerSLR3, which=1)
```

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




```
mplot(horsepowerSLR3, which=2)
```

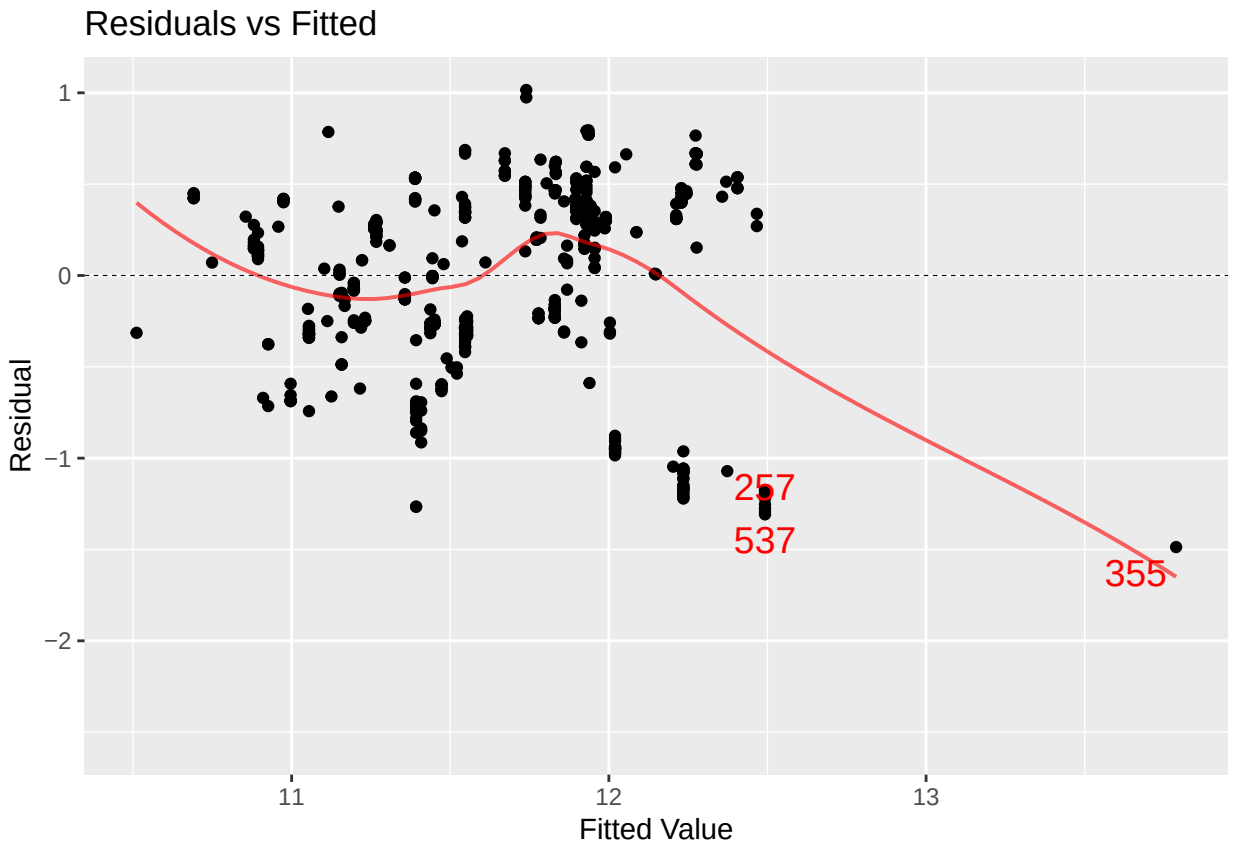



```
SportsCars <- mutate(SportsCars, logprice = log(price_in_usd))
horsepowerSLR2<- lm(logprice ~horsepower, data=SportsCars)
msummary(horsepowerSLR2)
```

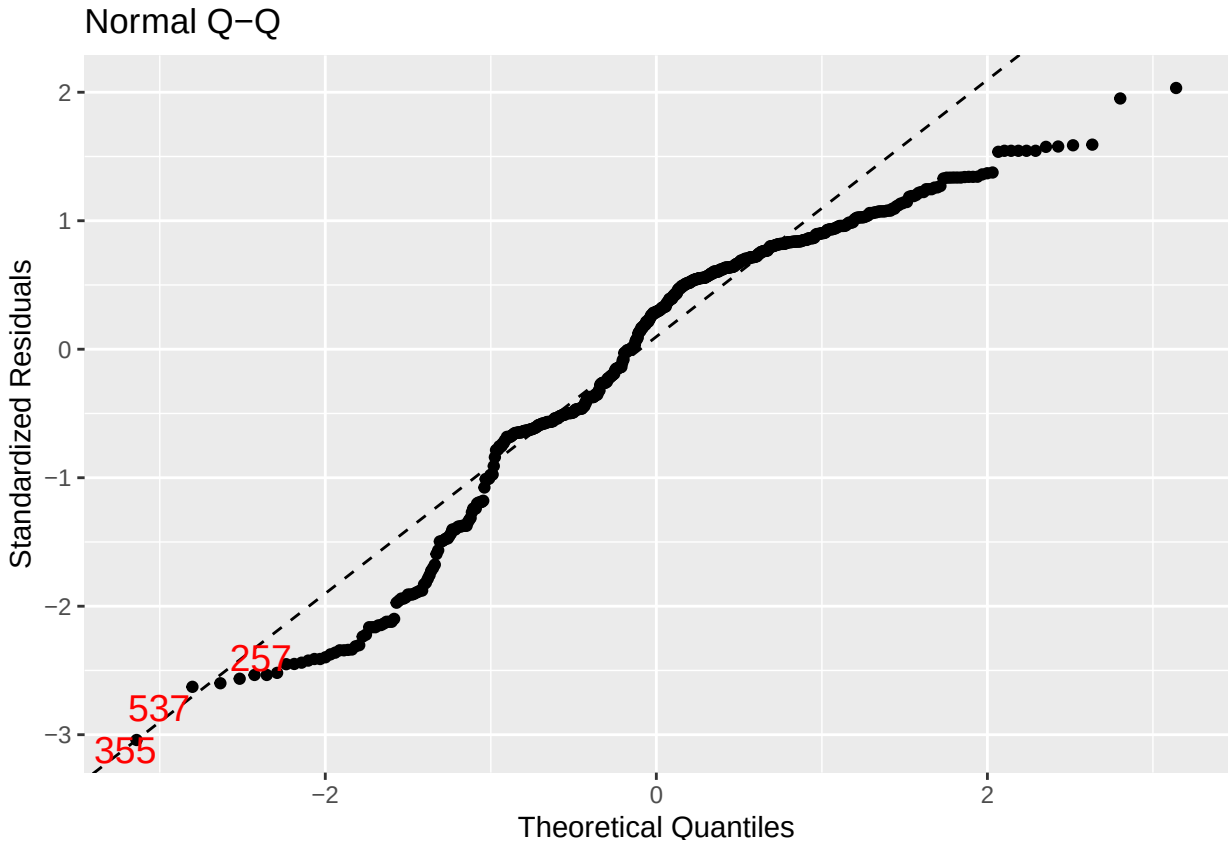
```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 9.929184   0.082960  119.69   <2e-16 ***
## horsepower  0.003216   0.000152   21.16   <2e-16 ***
##
## Residual standard error: 0.4999 on 590 degrees of freedom
## Multiple R-squared:  0.4314, Adjusted R-squared:  0.4304
## F-statistic: 447.6 on 1 and 590 DF,  p-value: < 2.2e-16
```

```
mplot(horsepowerSLR2, which=1)
```

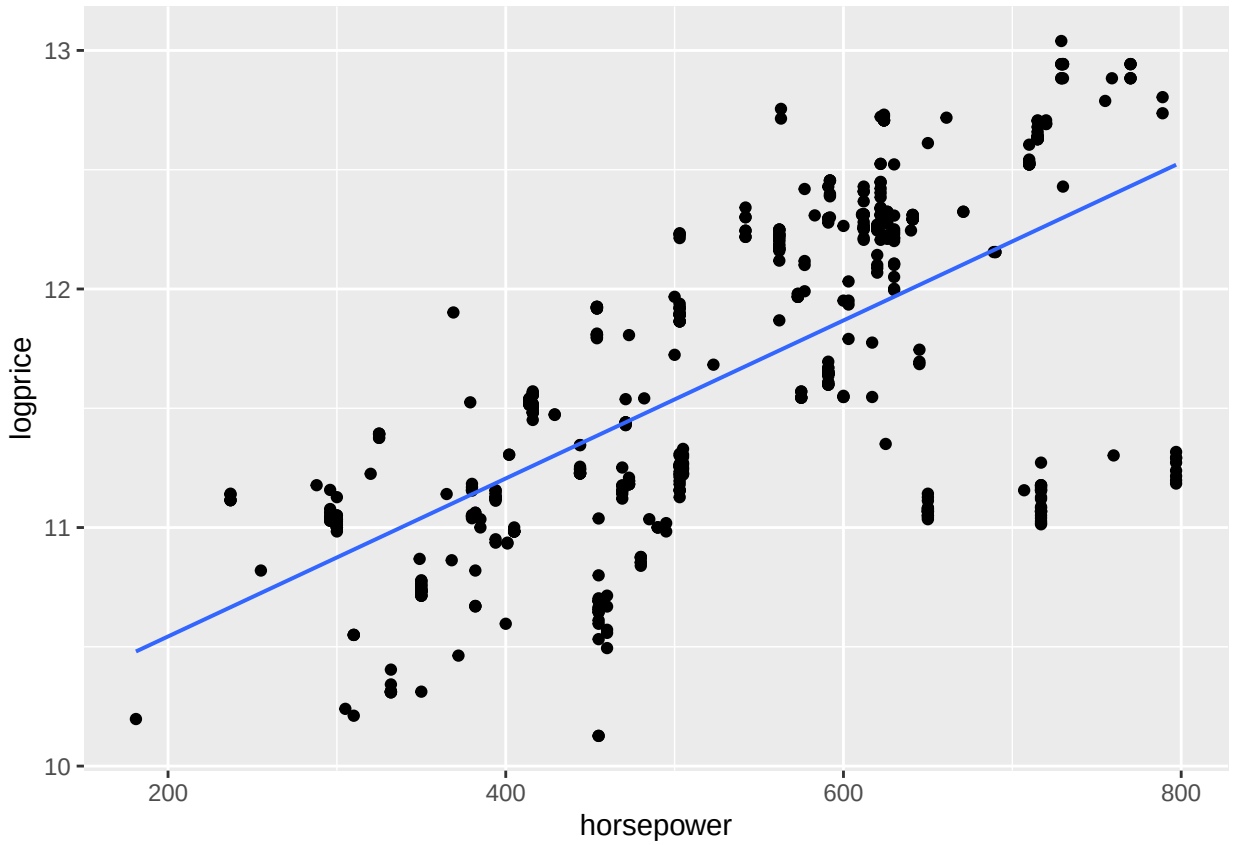
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
mplot(horsepowerSLR2, which=2)
```




```
SportsCars <- filter(SportsCars, torque_lb_ft<1299)
SportsCars <- mutate(SportsCars, logprice = log(price_in_usd))
gf_point(logprice ~horsepower, data=SportsCars) %>%
gf_lm()
```

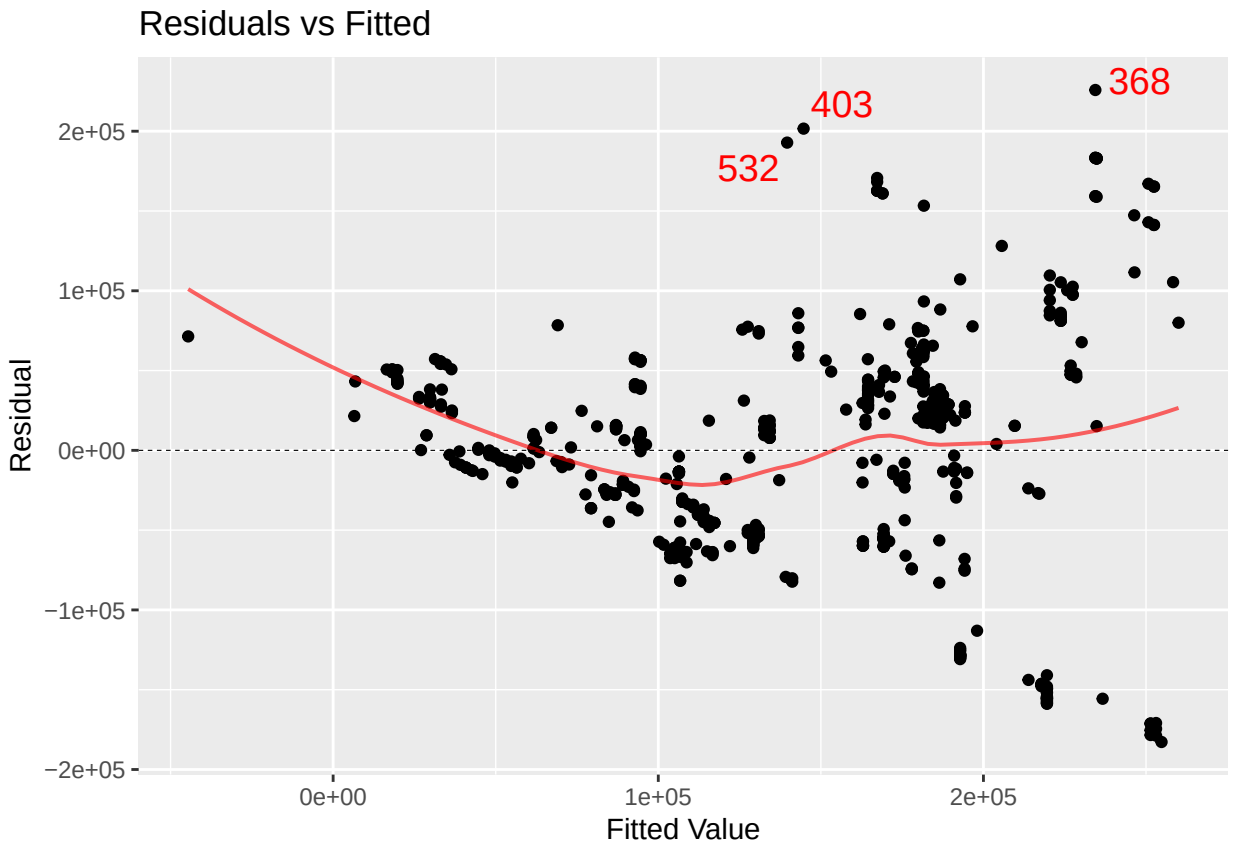



```
aovmultiplecars<- lm(price_in_usd ~x0_60_mph_time_seconds + horsepower, data=SportsCars)
msummary(aovmultiplecars)
```

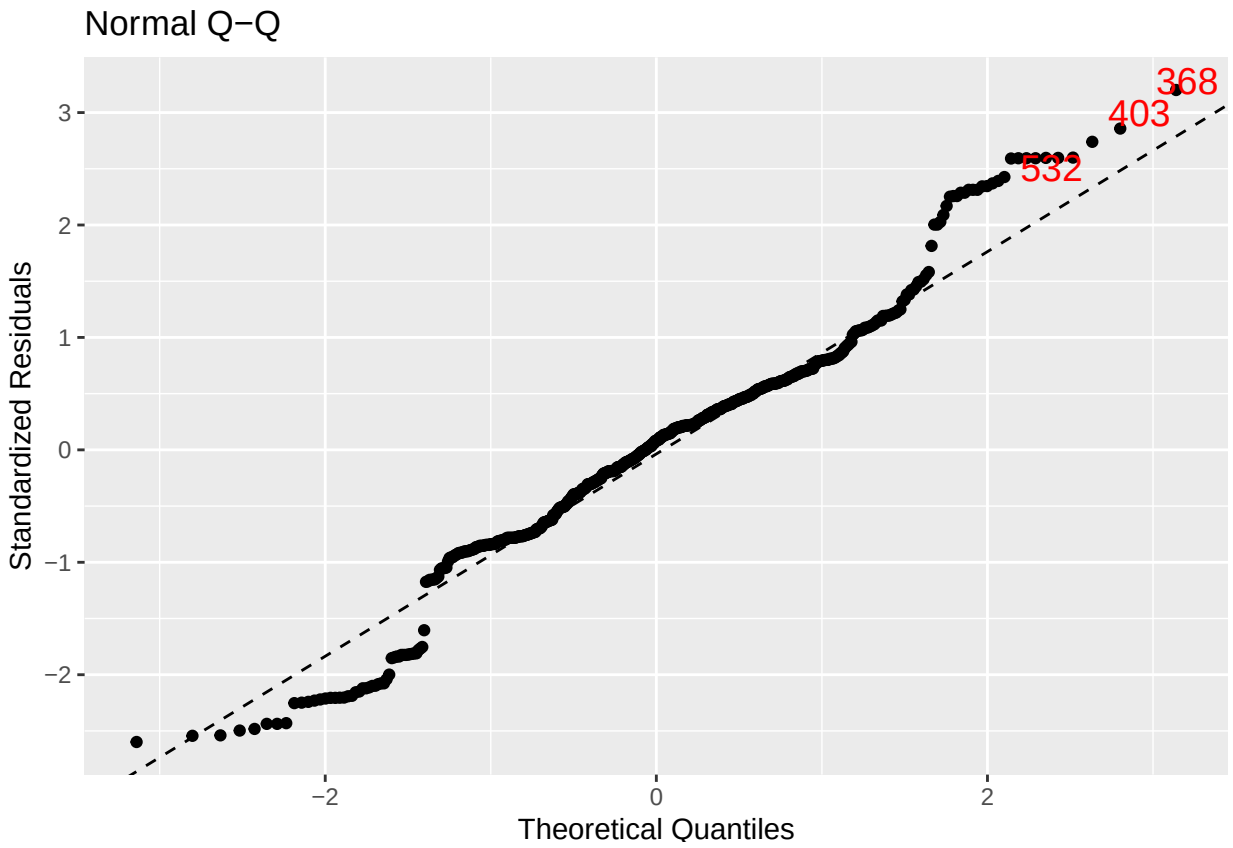
```
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -7160.2    41275.4  -0.173    0.862
## x0_60_mph_time_seconds -16851.3     6764.9  -2.491    0.013 *
## horsepower                398.4       34.4  11.582   <2e-16 ***
##
## Residual standard error: 70760 on 588 degrees of freedom
## Multiple R-squared:  0.4349, Adjusted R-squared:  0.433
## F-statistic: 226.2 on 2 and 588 DF,  p-value: < 2.2e-16
```

```
mplot(aovmultiplecars, which=1)
```

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




```
mplot(aovmultiplecars, which=2)
```

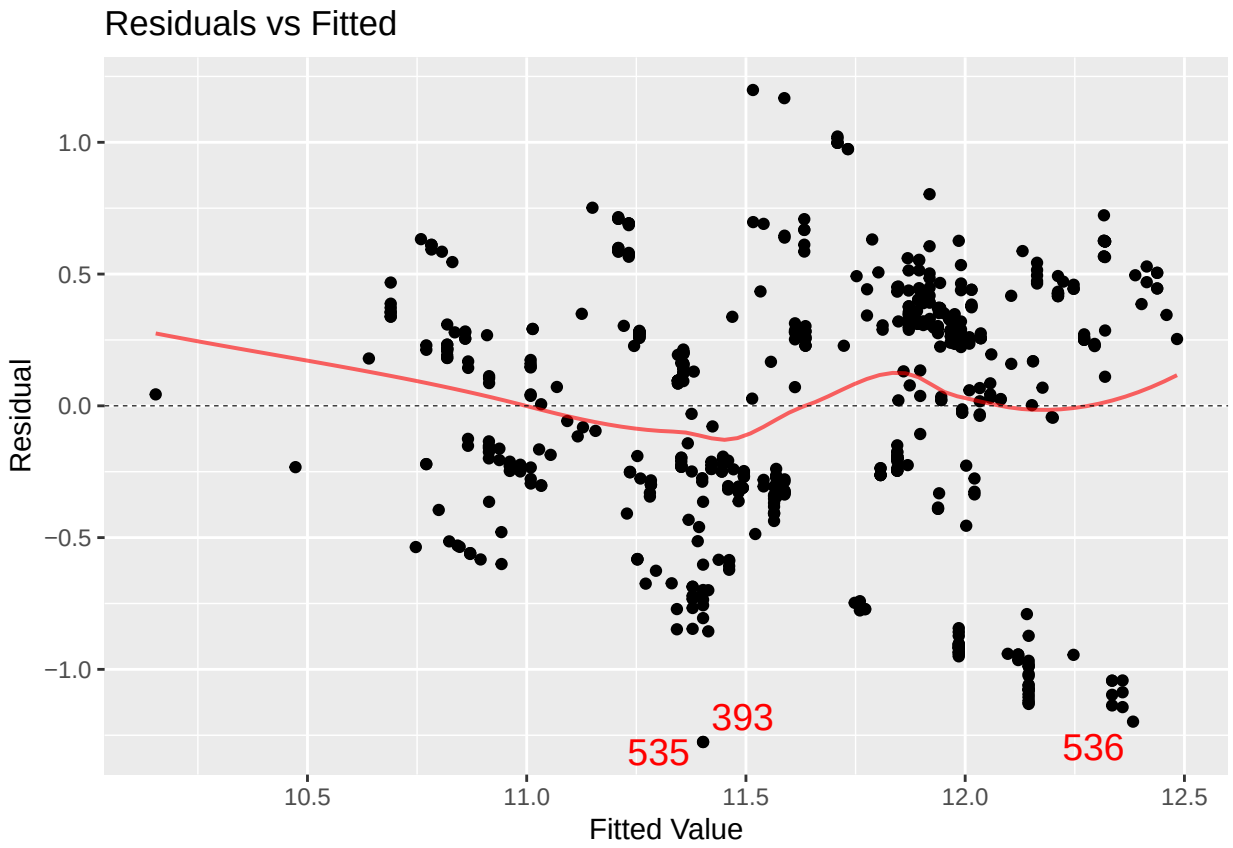



```
SportsCars<- mutate(SportsCars, logprice= log(price_in_usd))
aovbivariate2<-lm(logprice ~x0_60_mph_time_seconds + horsepower, data=SportsCars)
msummary(aovbivariate2)
```

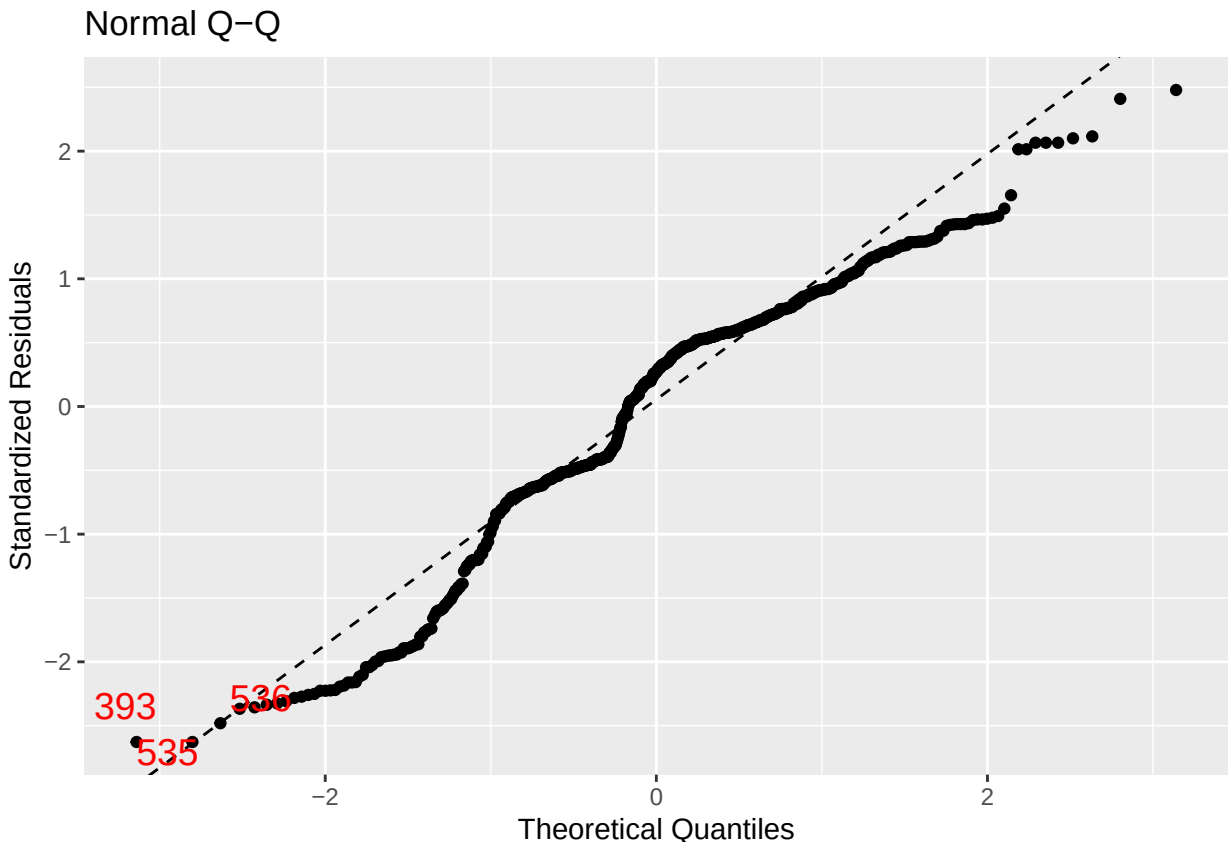
```
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            11.2744001  0.2835010  39.768  < 2e-16 ***
## x0_60_mph_time_seconds -0.2386388  0.0464647  -5.136 3.82e-07 ***
## horsepower              0.0023789  0.0002363  10.068  < 2e-16 ***
##
## Residual standard error: 0.486 on 588 degrees of freedom
## Multiple R-squared:  0.4634, Adjusted R-squared:  0.4616
## F-statistic: 253.9 on 2 and 588 DF,  p-value: < 2.2e-16
```

```
mplot(aovbivariate2, which=1)
```

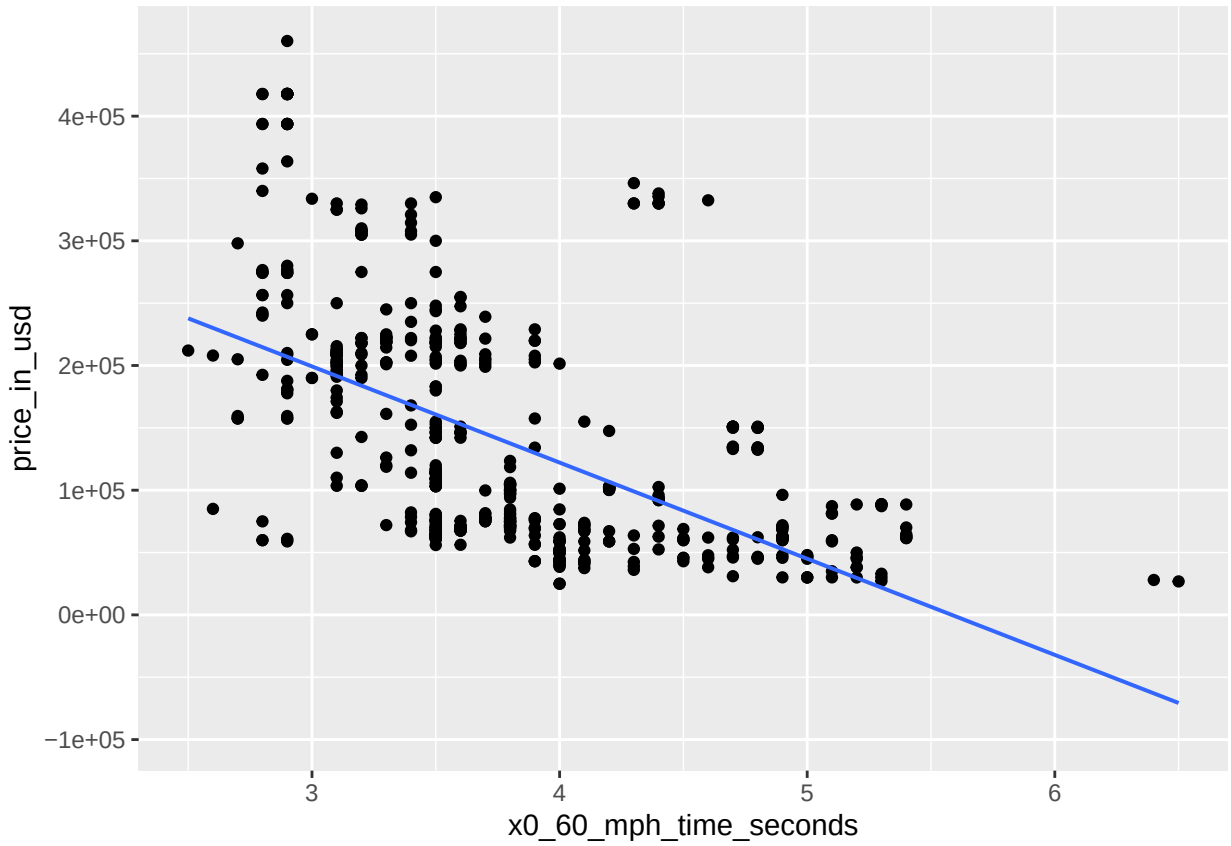
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
mplot(aovbivariate2, which=2)
```




```
gf_point(price_in_usd ~x0_60_mph_time_seconds, data=SportsCars) %>% gf_lm()
```

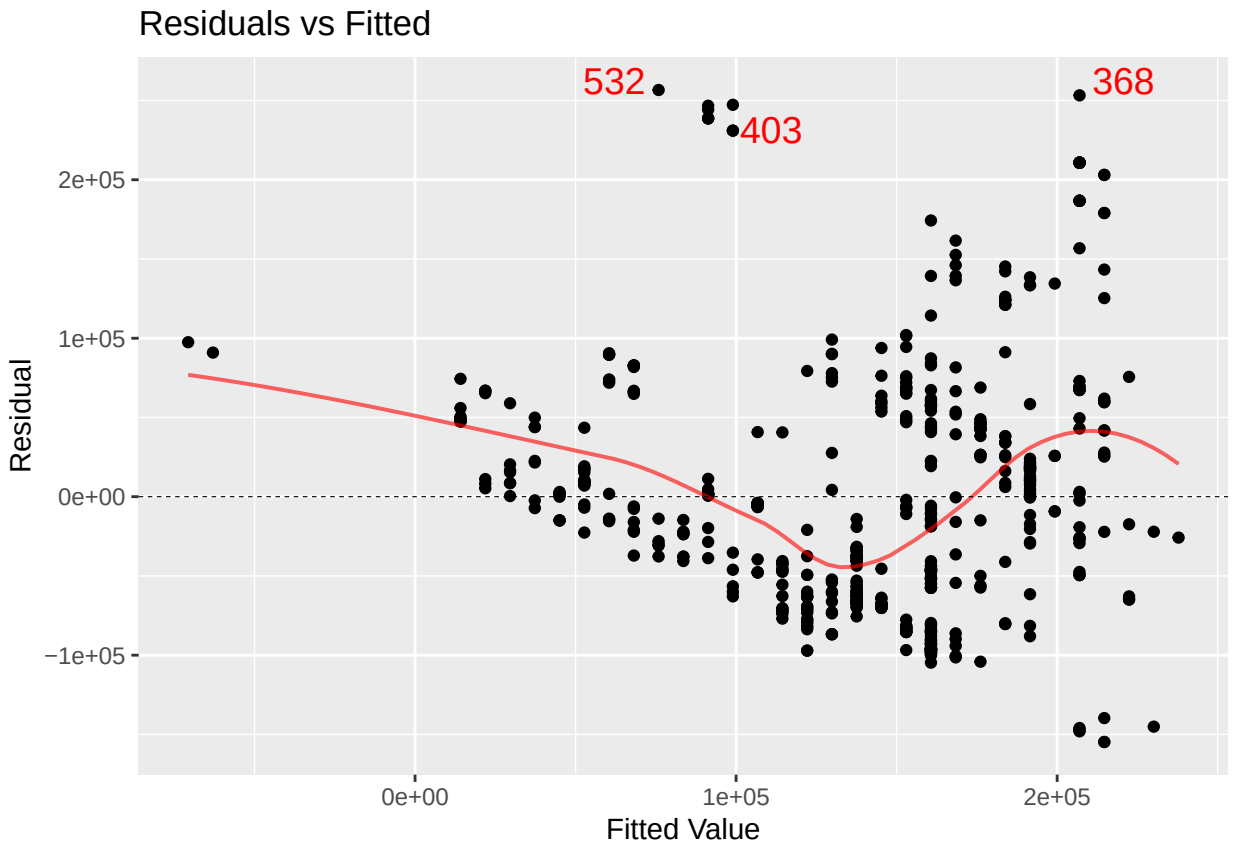



```
zerotosixtySLR<- lm(price_in_usd ~x0_60_mph_time_seconds, data=SportsCars)
msummary(zerotosixtySLR)
```

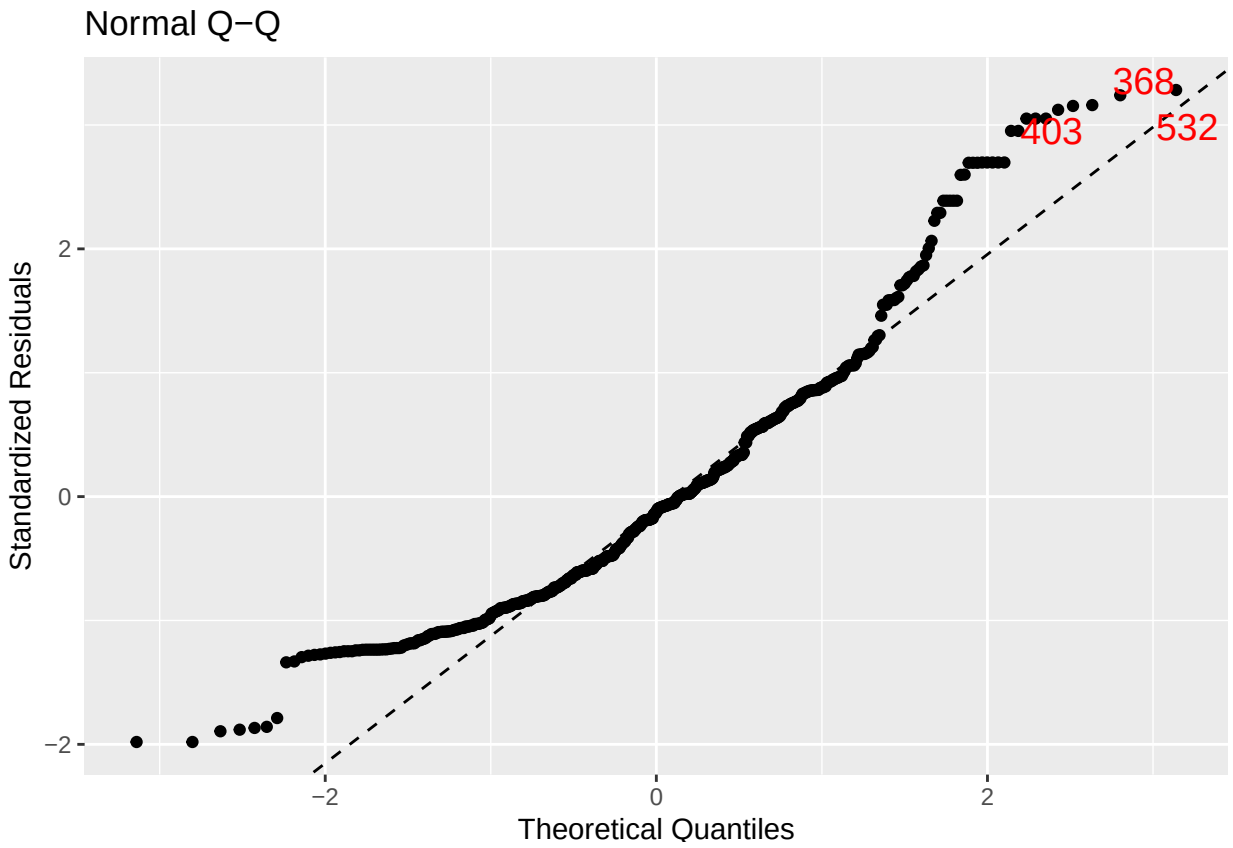
```
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)               430624      18356   23.46   <2e-16 ***
## x0_60_mph_time_seconds    -77121       4786  -16.11   <2e-16 ***
##
## Residual standard error: 78350 on 589 degrees of freedom
## Multiple R-squared:  0.306,  Adjusted R-squared:  0.3048
## F-statistic: 259.7 on 1 and 589 DF,  p-value: < 2.2e-16
```

```
mplot(zerotosixtySLR, which=1)
```

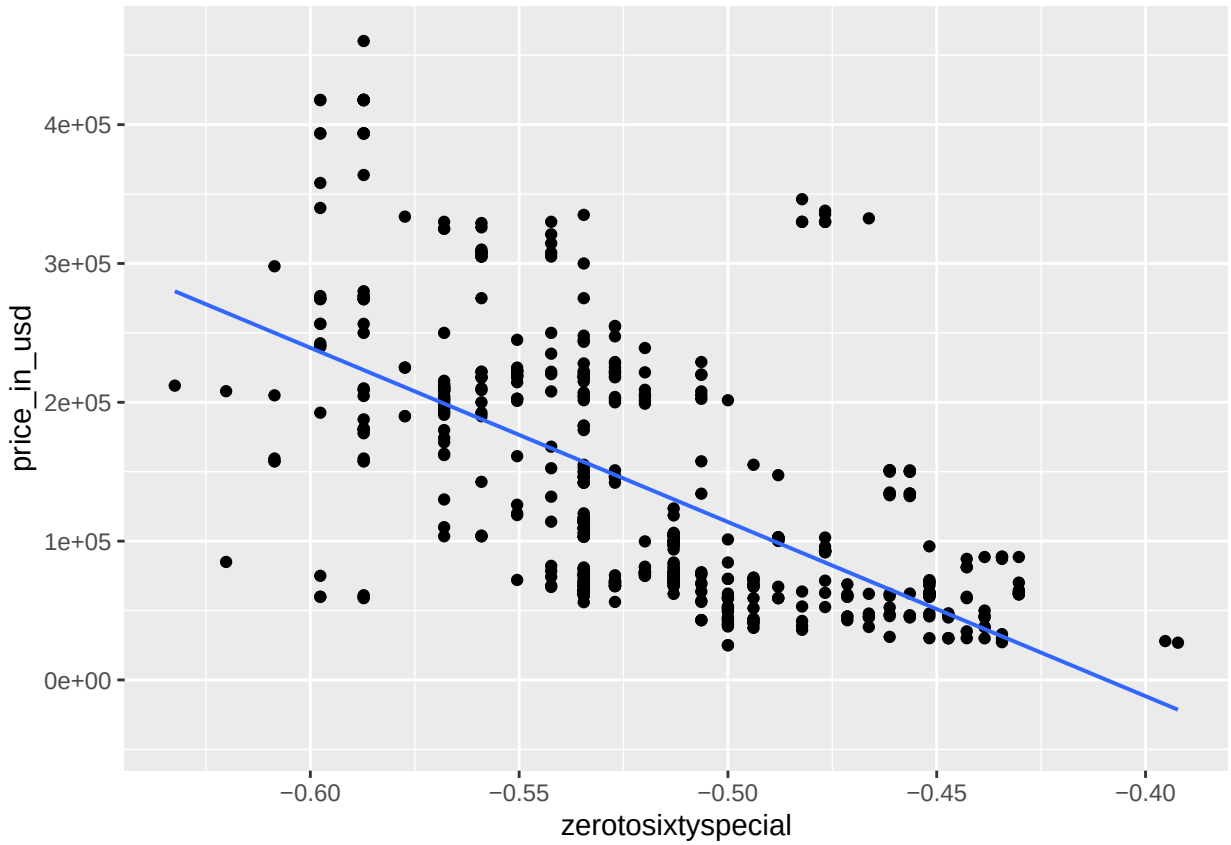
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
mplot(zerotosixtySLR, which=2)
```




```
SportsCars <- mutate(SportsCars, zerotosixtyspecial = -1/sqrt(x0_60_mph_time_seconds))
gf_point(price_in_usd ~ zerotosixtyspecial, data=SportsCars) %>%
gf_lm()
```

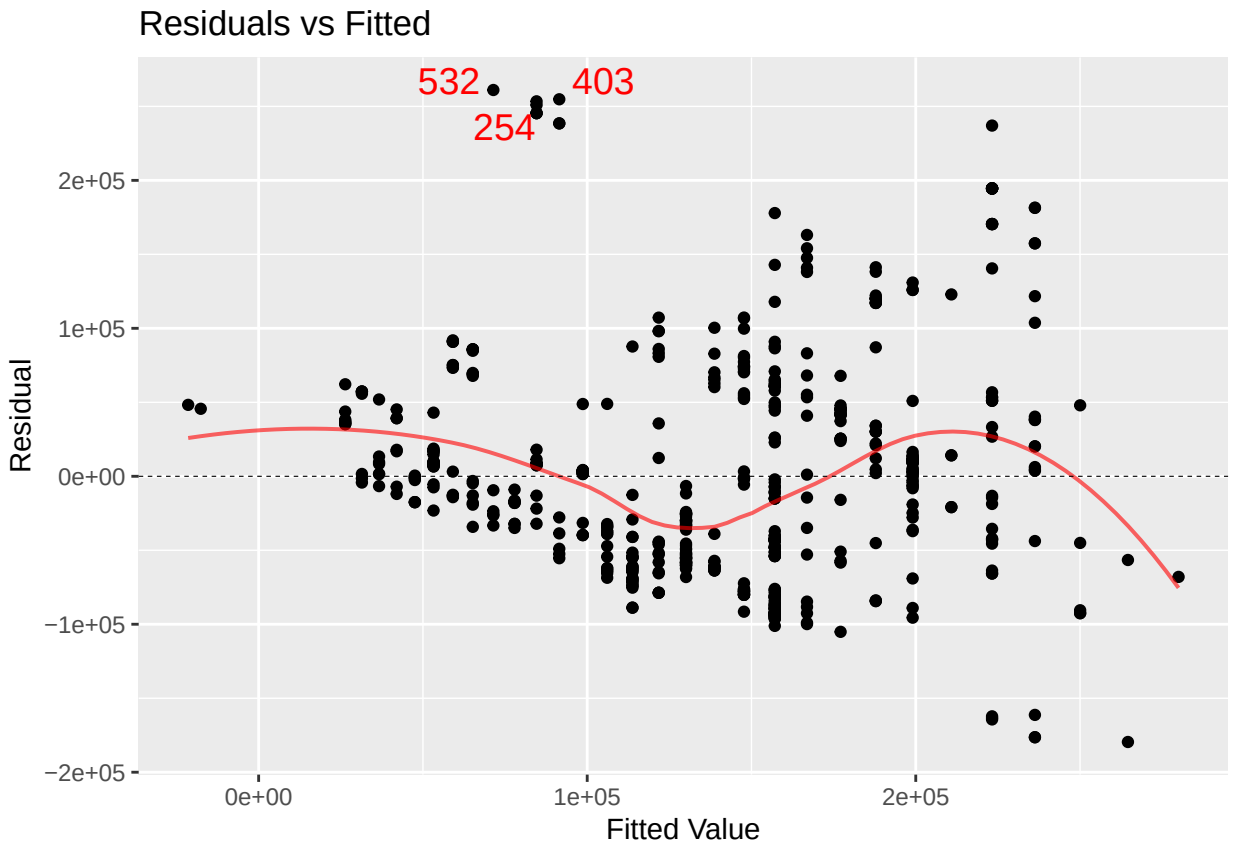



```
zerotosixtySLR3<- lm(price_in_usd ~ zerotosixtyspecial, data=SportsCars)
msummary(zerotosixtySLR3)
```

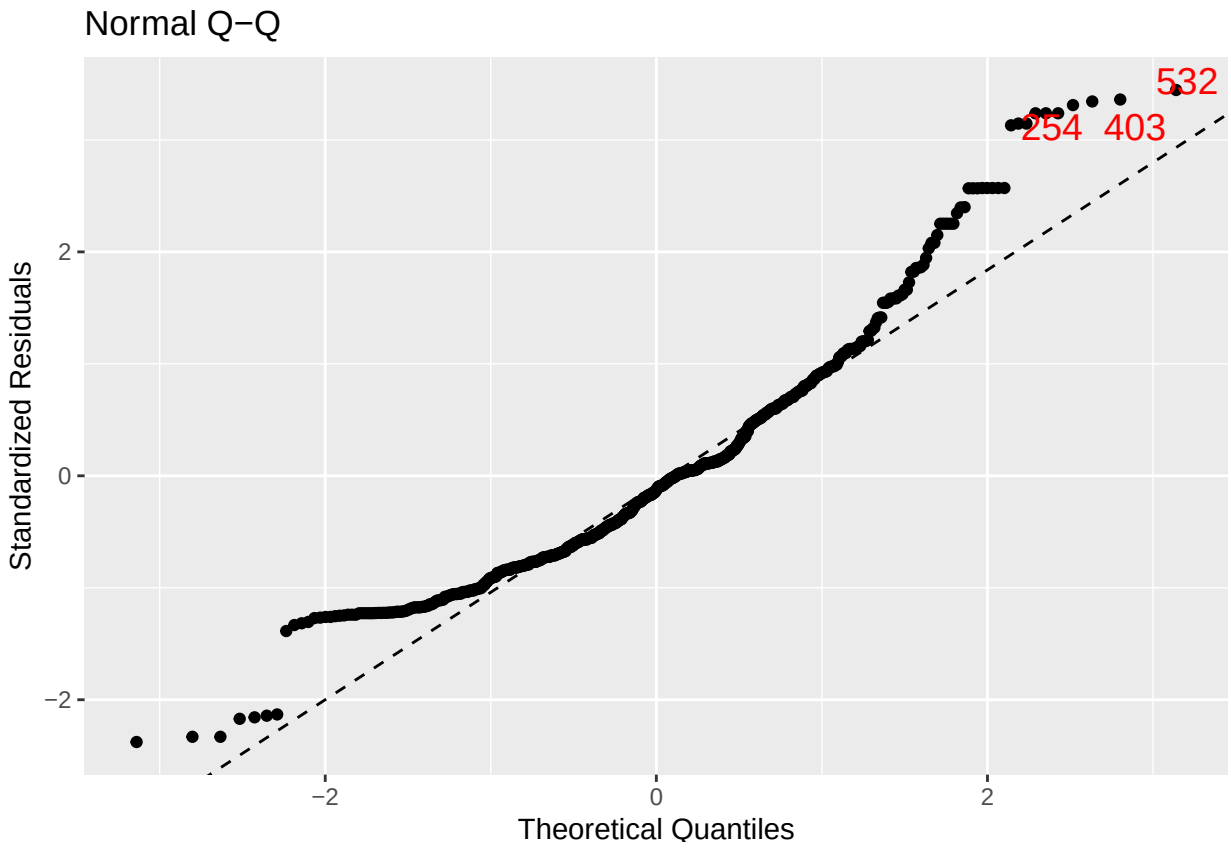
```
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -513628      36948  -13.90   <2e-16 ***
## zerotosixtyspecial -1254813      70740  -17.74   <2e-16 ***
##
## Residual standard error: 75920 on 589 degrees of freedom
## Multiple R-squared:  0.3482, Adjusted R-squared:  0.3471
## F-statistic: 314.7 on 1 and 589 DF,  p-value: < 2.2e-16
```

```
mplot(zerotosixtySLR3, which=1)
```

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




```
mplot(zerotosixtySLR3, which=2)
```




```
kitchen1<- lm(price_in_usd ~ year + engine_size_l + horsepower + torque_lb_ft + x0_60_mph_time_seconds,
msummary(kitchen1)
```

```
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              1.362e+07  5.662e+06   2.405  0.01648 *
## year                    -6.768e+03  2.802e+03  -2.416  0.01600 *
## engine_size_l           -9.482e+03  3.003e+03  -3.158  0.00167 **
## horsepower               8.772e+02  5.892e+01  14.889  < 2e-16 ***
## torque_lb_ft            -4.370e+02  4.635e+01  -9.429  < 2e-16 ***
## x0_60_mph_time_seconds -3.324e+03  6.677e+03  -0.498  0.61880
##
## Residual standard error: 64850 on 585 degrees of freedom
## Multiple R-squared:  0.5276, Adjusted R-squared:  0.5236
## F-statistic: 130.7 on 5 and 585 DF,  p-value: < 2.2e-16
```

```
car::vif(kitchen1) #x_60_mph_time_seconds not significant
```

```
##                   year          engine_size_l             horsepower
##               1.137823               2.239846               8.552275
##           torque_lb_ft x0_60_mph_time_seconds
##               4.798533               2.840297
```

```
stepwise <- regsubsets(price_in_usd ~ year + engine_size_l + horsepower + x0_60_mph_time_seconds, data =
with(summary(stepwise), data.frame(cp, outmat)) #scrapped torque_lb_ft because of high correlation with
```

```
##                  cp year engine_size_l horsepower x0_60_mph_time_seconds
## 1  ( 1 )  27.990964                            *
## 2  ( 1 ) 341.749712    *             *
## 3  ( 1 )   4.347689    *             *          *
```

```
## 4  ( 1 )   5.000000    *            *          *                      *
```

```
favmod<- lm(price_in_usd ~ year + engine_size_l + horsepower, data=SportsCars)
msummary(favmod)
```

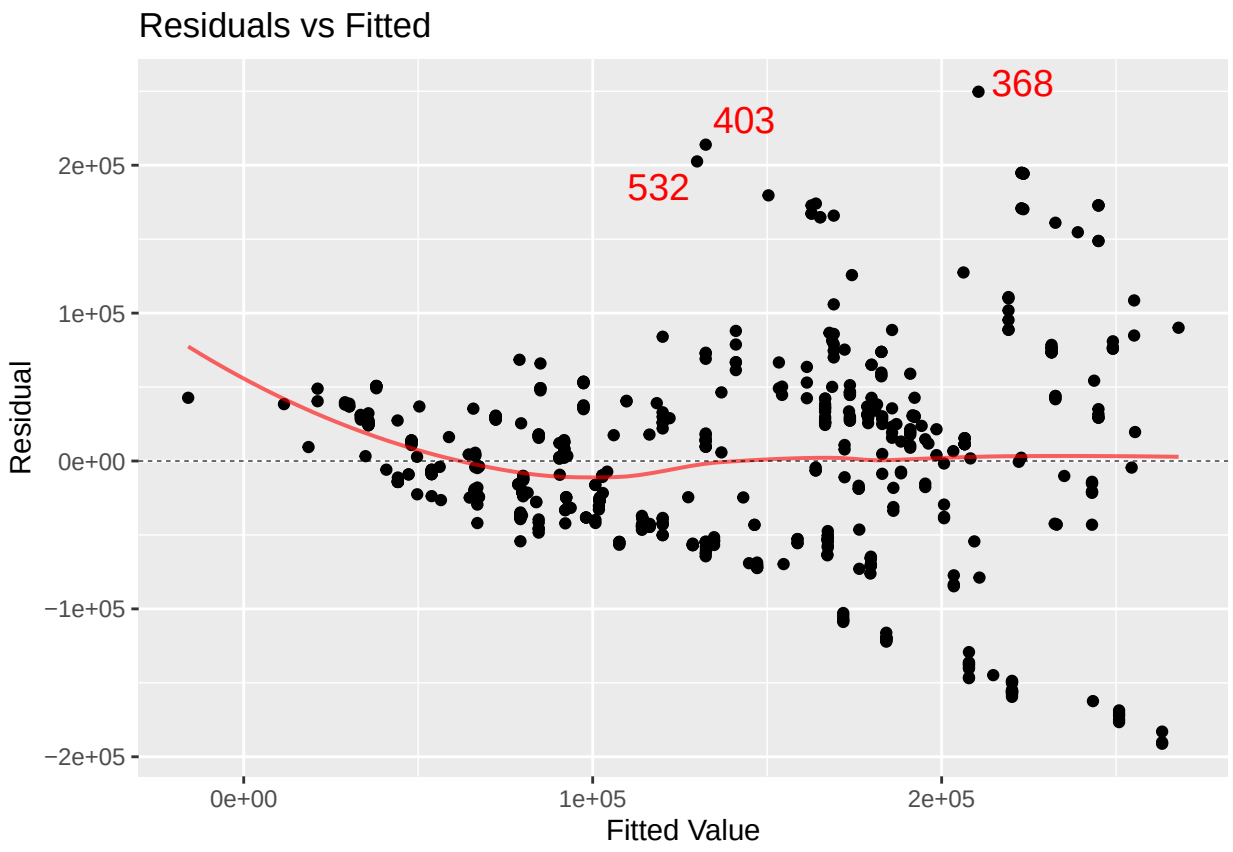
```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.484e+07  5.932e+06   4.187 3.26e-05 ***
## year          -1.233e+04  2.934e+03  -4.204 3.04e-05 ***
## engine_size_l -1.241e+04  2.997e+03  -4.141 3.97e-05 ***
## horsepower     5.377e+02  2.919e+01  18.417  < 2e-16 ***
##
## Residual standard error: 69570 on 587 degrees of freedom
## Multiple R-squared:  0.4546, Adjusted R-squared:  0.4518
## F-statistic: 163.1 on 3 and 587 DF,  p-value: < 2.2e-16
```

```
car::vif(favmod) #best model according to stepwise
```

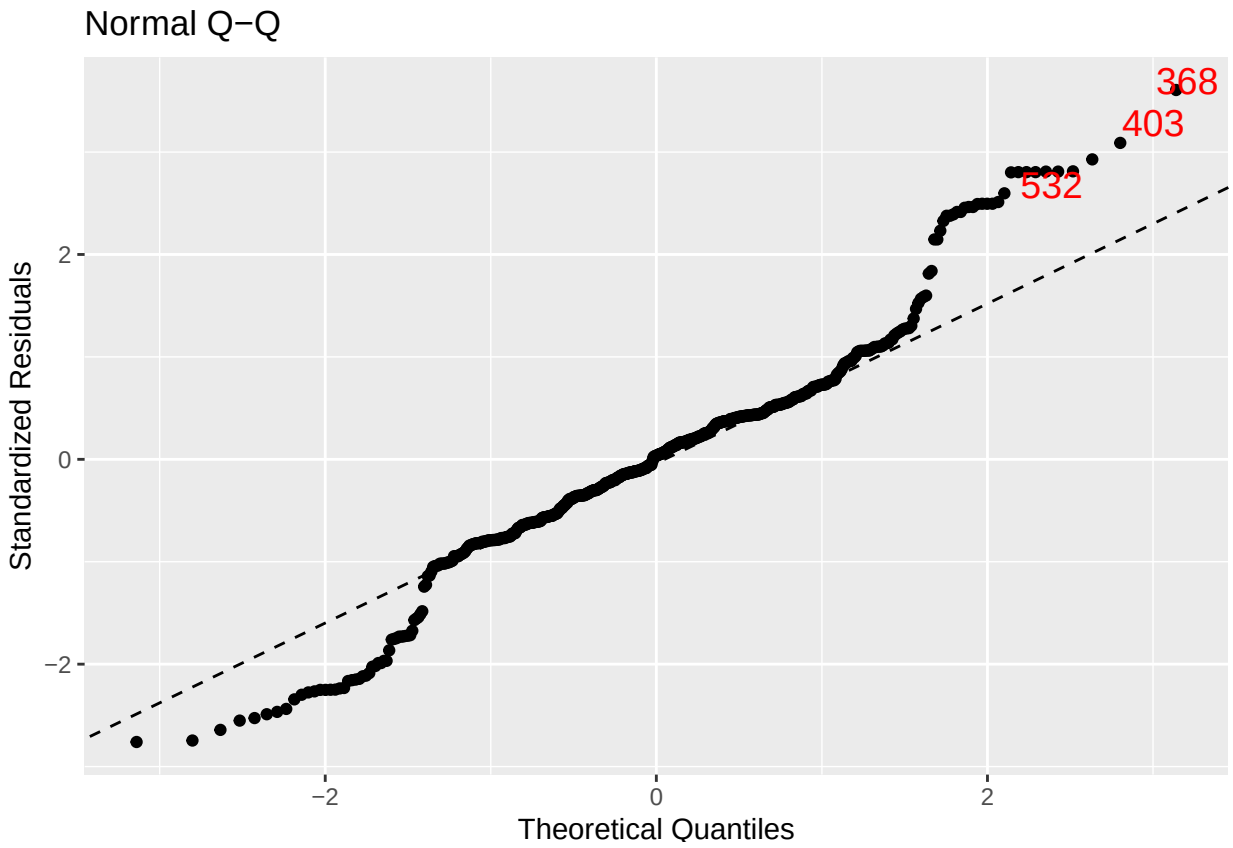
```
##          year engine_size_l    horsepower
##      1.084608      1.938233      1.824585
```

```
mplot(favmod, which=1)
```

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

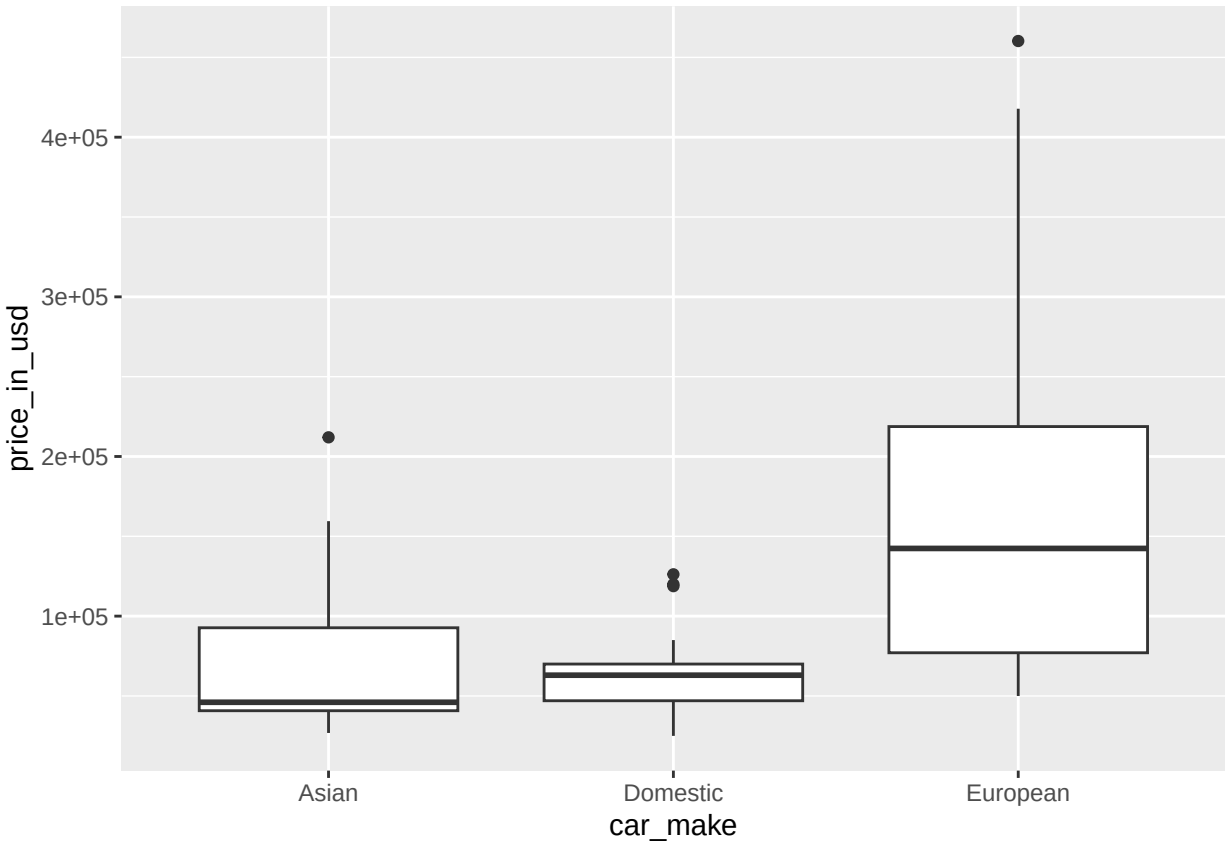



```
mplot(favmod, which=2)
```



Next, we will attempt to run an ANOVA with the factor car_make, or the car region of origin. The results will allow us to determine if the geographical region of production affects the price enough to add it to a potential multivariate model. This factor has three categories: Asian, European, and Domestic. To get a better sense of this categorical variable, we compared box plots of each category. Immediately, we see that the European cars have a much higher standard deviation than both other categories, but especially domestic cars. Although this could be due to smaller sample sizes of Domestic and Asian car brands, in order to compare the categories at all, we need to check the requisite conditions to run an ANOVA.

```
gf_boxplot(price_in_usd ~ car_make, data = SportsCars)
```




```
location <- lm(price_in_usd ~ car_make, data = SportsCars)
favstats(price_in_usd ~ car_make, data = SportsCars)
```

```
##   car_make   min      Q1 median       Q3    max      mean       sd   n missing
## 1    Asian 26830 40750.0  46050  92712.5 212000  66062.41 44474.55  54       0
## 2 Domestic 25000 47000.0  63000  69995.0 126190  60880.75 18167.74  87       0
## 3 European 50000 77062.5 142400 218750.0 460247 163421.82 94260.05 450       0
```

We then checked the conditions to run an ANOVA. The condition of equal variance is clearly not passed, due to the largest over smallest standard deviation being well over 2 ($94260.05/18167.74 = 5.188$) and the obvious lack of continuity on the Residuals vs. Fitted plot. Furthermore, the condition of normality is not passed due to the tails of the QQ plot clearly seperating from the line.

```
mplot(location, which = 1)
```

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

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 60368
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 1.0305e+05
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 1.0408e-15
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 1.6393e+05
```

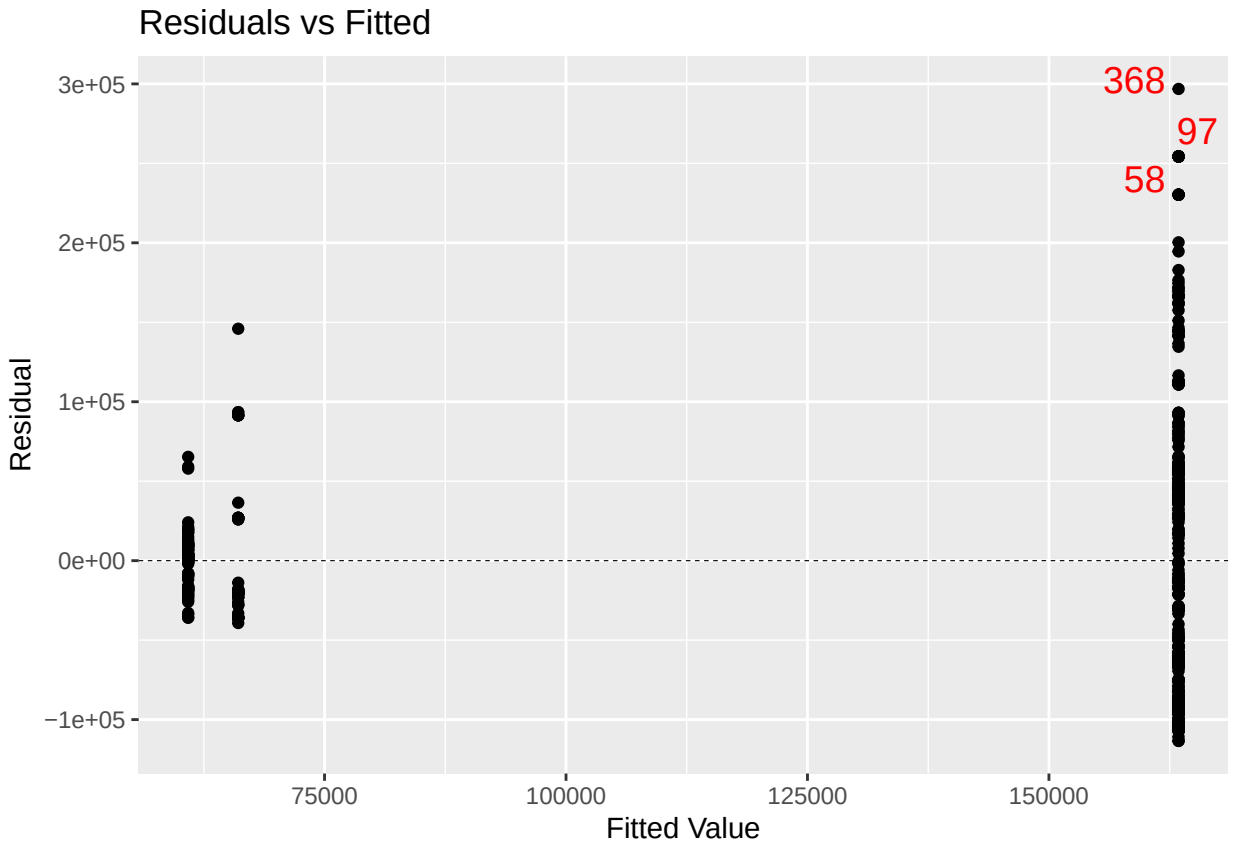
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 2.6287e+05
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger
```

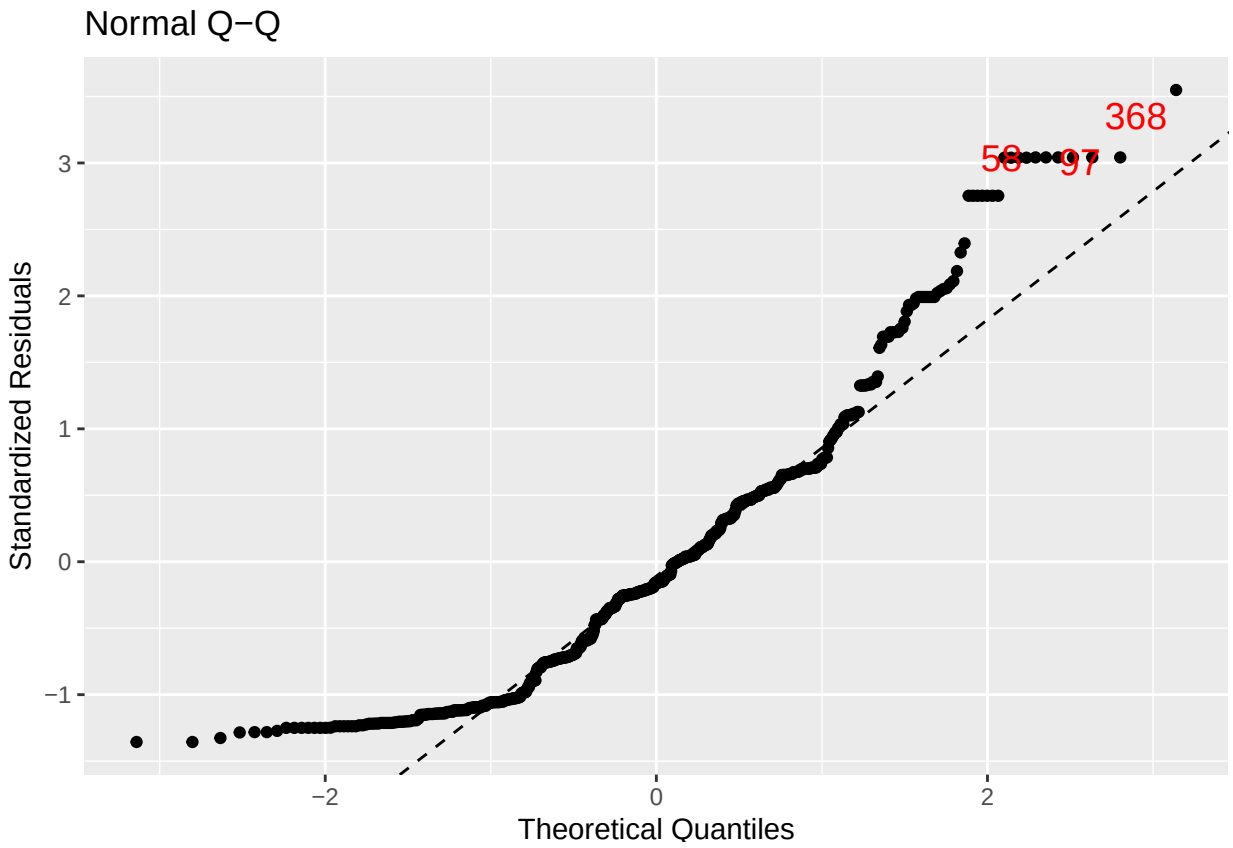
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 2.6287e+05
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger
```

```
## Warning: Failed to fit group -1.
## Caused by error in `predLoess()`:
## ! NA/NaN/Inf in foreign function call (arg 5)
```

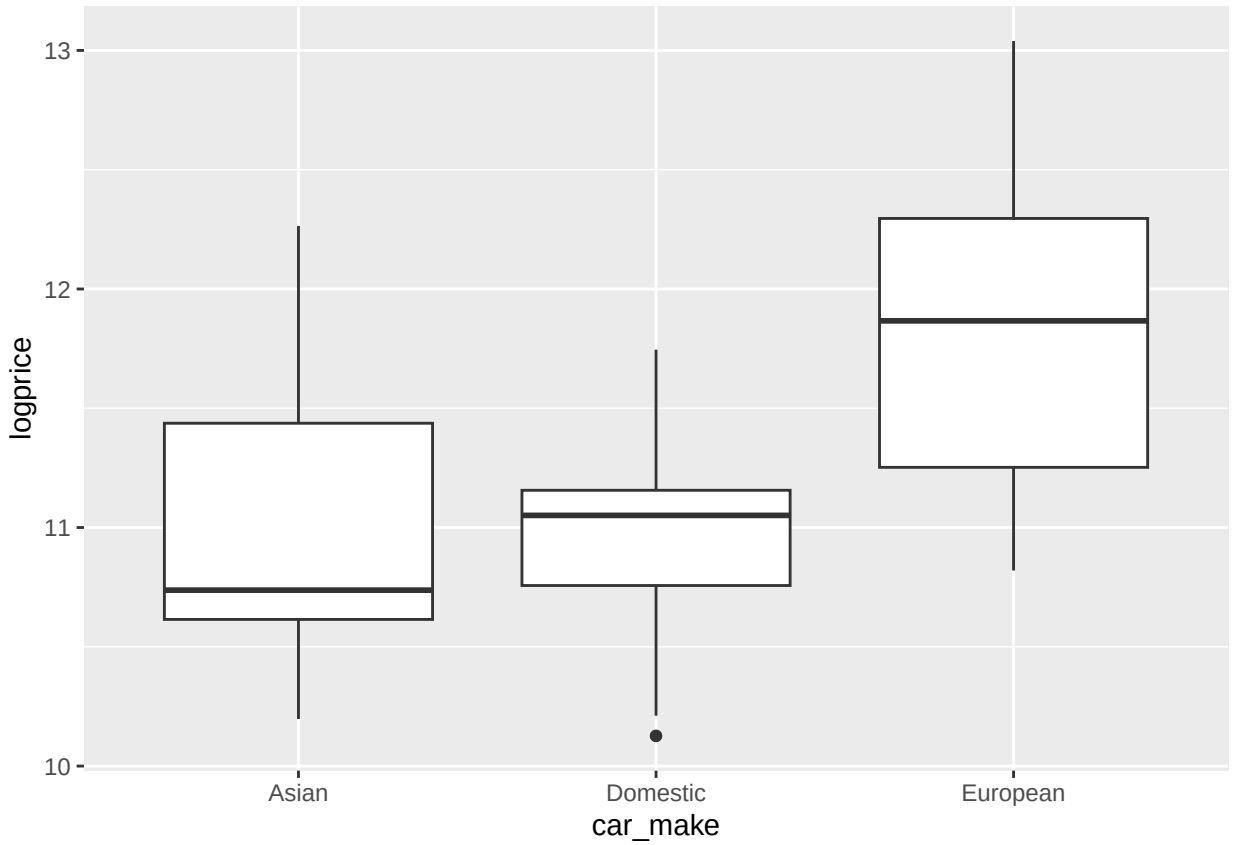



```
mplot(location, which = 2)
```



Since we did not pass conditions and still want to run a comparison test on this variable, the next step of action is to attempt a transformation. We attempted to take the natural log of the response variable, price. Although this helped increase our equal variance and pass the standard deviation test ($0.579/0.308 = 1.880$), we still see a clear lack of normality in the QQ plot.

```
gf_boxplot(logprice ~ car_make, data = SportsCars)
```




```
loglocation <- lm(logprice ~ car_make, data = SportsCars)
favstats(logprice ~ car_make, data = SportsCars)
```

```
##   car_make      min       Q1   median       Q3      max     mean        sd   n
## 1    Asian 10.19728 10.61471 10.73748 11.43725 12.26434 10.92960 0.5499697  54
## 2 Domestic 10.12663 10.75700 11.05089 11.15618 11.74554 10.97184 0.3081777  87
## 3 European 10.81978 11.25237 11.86639 12.29568 13.03952 11.83934 0.5793527 450
##   missing
## 1       0
## 2       0
## 3       0
```

```
mplot(loglocation, which = 1)
```

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

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 10.925
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 0.91429
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 7.7306e-16
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : at 11.844
```

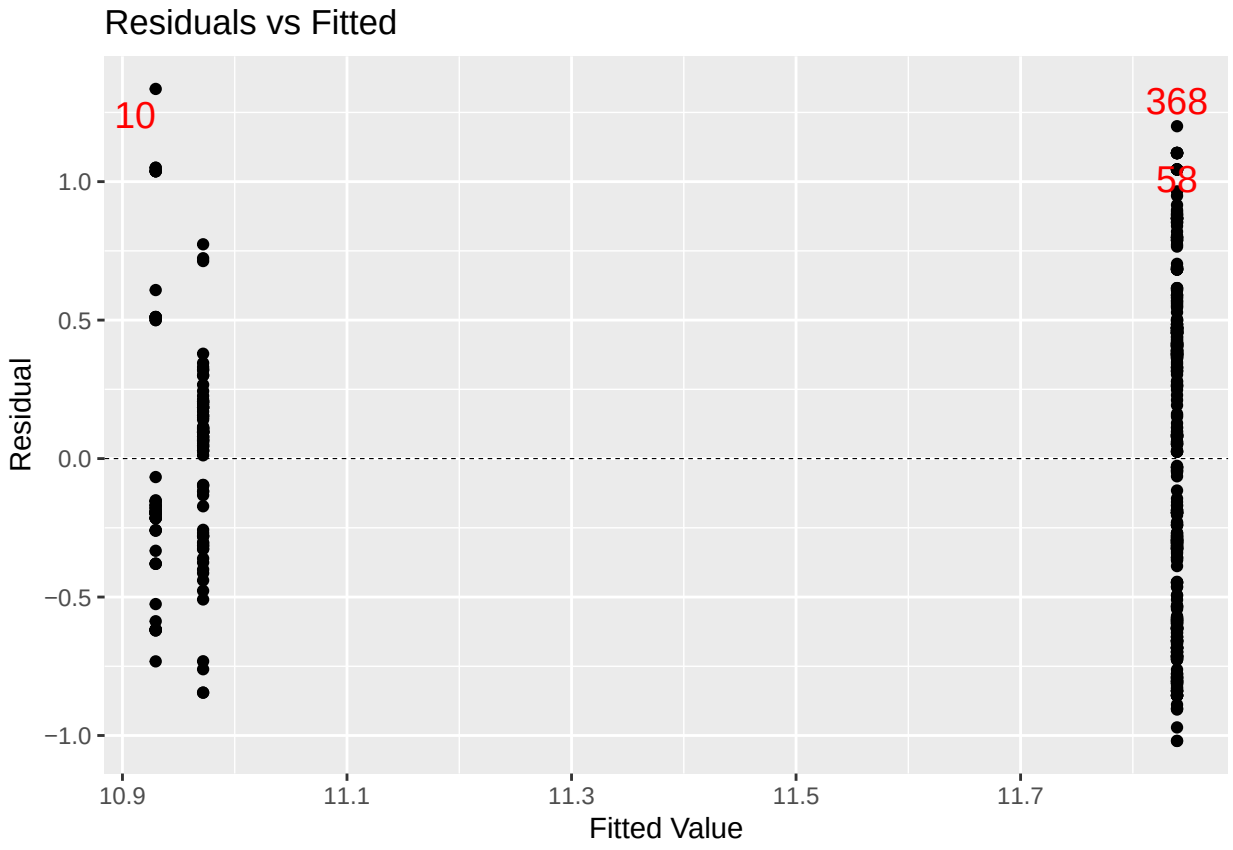
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : radius 2.0691e-05
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : all data on boundary of neighborhood. make span bigger
```

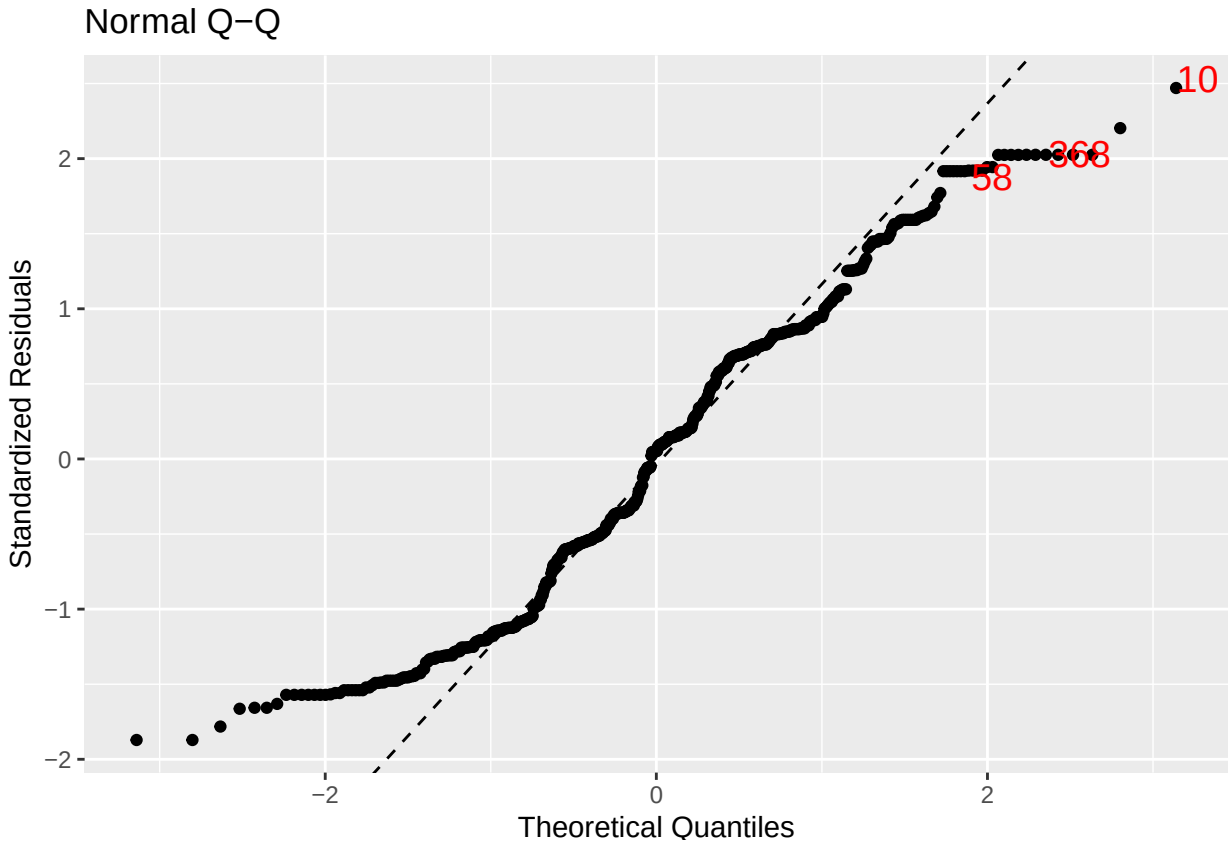
```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 2.0691e-05
```

```
## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : zero-width neighborhood. make span bigger
```

```
## Warning: Failed to fit group -1.
## Caused by error in `predLoess()`:
## ! NA/NaN/Inf in foreign function call (arg 5)
```




```
mplot(loglocation, which = 2)
```



Since we could not find a transformation that allowed us to pass conditions for running an ANOVA, we next moved towards running a randomization F-test. This test can help us compare categories across a response variable even when we do not pass conditions for an ANOVA. First, we randomly shuffled the price 10000 times in order to compare the frequency at which a random shuffle (which should have no main effects) will take on the same F-value as we observe in this data set.

```
set.seed(40)
randomizedsamples <- do(10000) * (anova(lm(shuffle(price_in_usd) ~ car_make, data = SportsCars))$"F valu
randomizedsamples <- as.data.frame(randomizedsamples)
```

We then calculated the baseline F value, which is 77.48.

```
anova(location)
```

```
## Analysis of Variance Table
##
## Response: price_in_usd
##            Df     Sum Sq    Mean Sq F value    Pr(>F)
## car_make    2 1.0865e+12 5.4324e+11  77.482 < 2.2e-16 ***
## Residuals 588 4.1226e+12 7.0112e+09
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

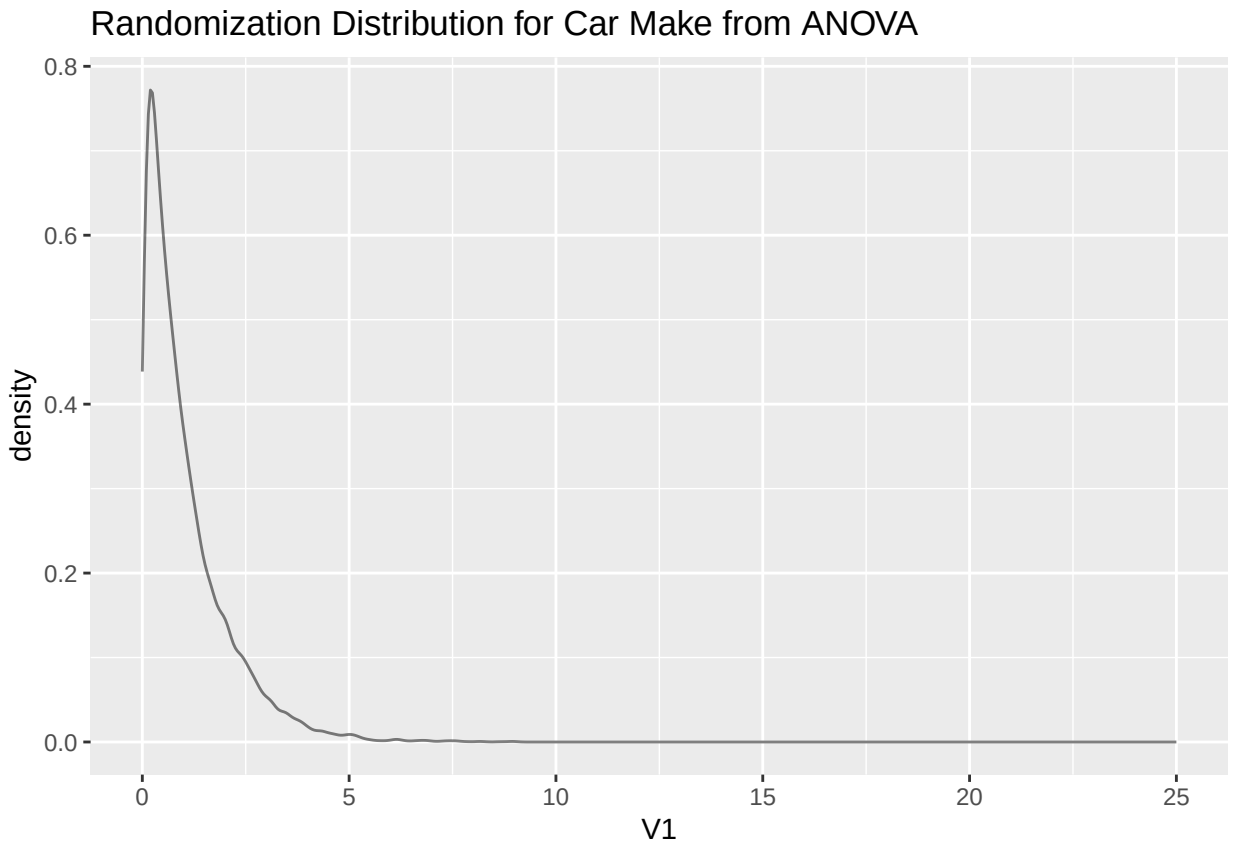
```
origres <- anova(location)$"F value"#
origres
```

```
## [1] 77.48239       NA
```

We then will observe how often our randomly sampled sets end up with an F value close to the 77.48 we see in our original data set. We observe that the random samples do not come close to obtaining an F value that

high. In fact, we see almost no F-values over 5.

```
gf_dens(~ V1, data = randomizedsamples) %>%
  gf_lims(x = c(0, 25)) %>%
  gf_labs(title = "Randomization Distribution for Car Make from ANOVA")
```



Next, we see the probability of getting an F like ours when the null (sports cars produced everywhere have the same price) is true. This value is 0, indicating that we should reject the null. We have evidence, using F-randomization, that there is a significant difference in the average price of a sports car between at least one of these three geographical regions.

```
pdata(~ V1, origres[1], data = randomizedsamples, lower.tail = FALSE)
```

```
## [1] 0
```

Since we got a significant result from the randomization F-test, we then will run a post-Hoc test to determine which categories are significantly different. We ran Tukey's HSD and found evidence that there are significant differences in the average price of sports cars that are produced in Europe vs. Asia ($p < 0.001$) and significant differences in the average price of cars produced domestically vs. those produced in Europe ($p < 0.001$).

```
TukeyHSD(location)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = x)
##
## $car_make
##                        diff       lwr       upr     p adj
## Domestic-Asian     -5181.66 -39265.87  28902.55 0.9320949
```

```
## European-Asian     97359.41  69025.13 125693.70 0.0000000
## European-Domestic 102541.08  79499.01 125583.14 0.0000000
```

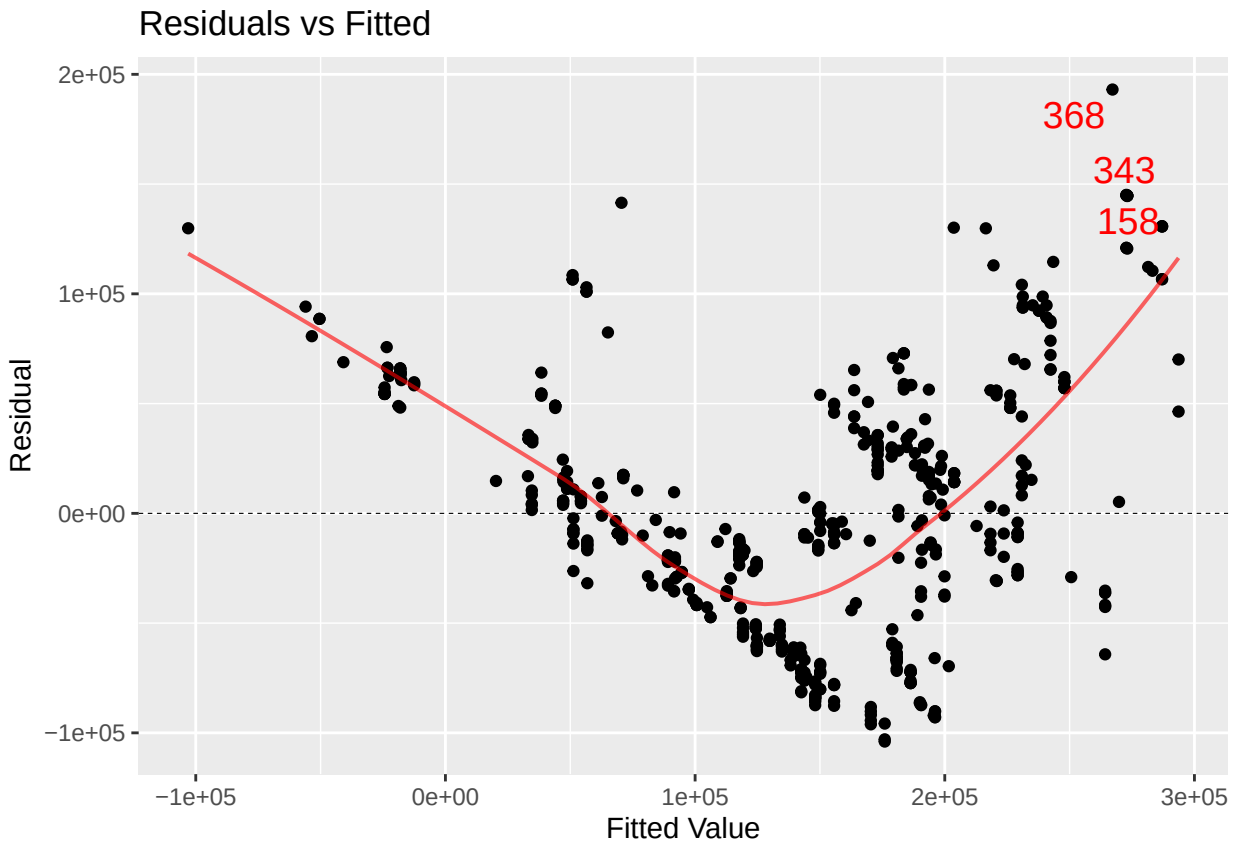
Since there is evidence that European cars are significantly more expensive than Asian or Domestic makes, we will attempt to add an indicator variable to our best multiple linear regression model to determine if it reduces the error. By indicating all European cars with a "1" and all others with a "0", we can add the variable to our model.

```
SportsCars <- SportsCars %>% mutate(car_make = case_when(
    car_make %in% c("Asian") ~ 0,
    car_make %in% c("Domestic") ~ 0,
    car_make %in% c("European") ~ 1))
MLR5<- lm(price_in_usd ~ year + engine_size_l + horsepower + car_make, data=SportsCars)
msummary(MLR5)
```

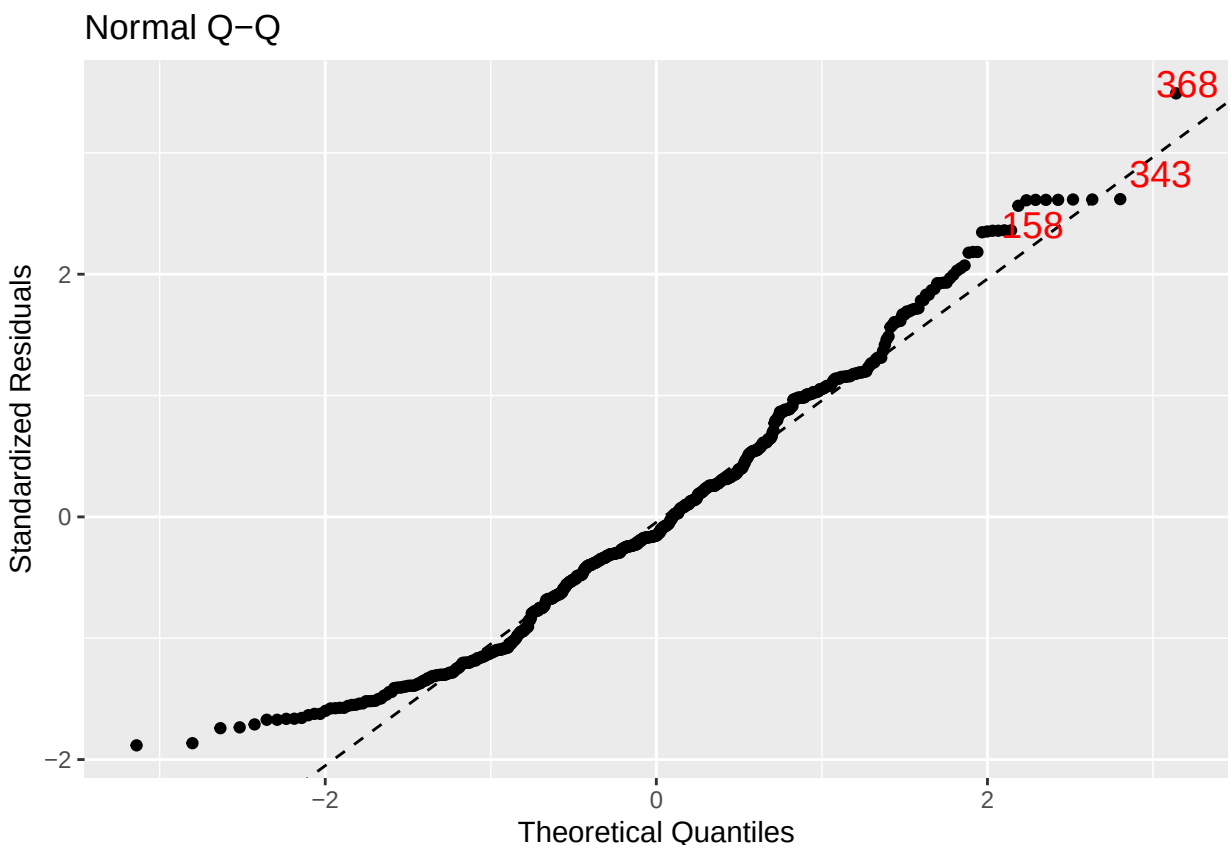
```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.104e+07  4.807e+06   2.297   0.0220 *
## year          -5.559e+03  2.377e+03  -2.339   0.0197 *
## engine_size_l  1.534e+04  2.841e+03   5.399 9.75e-08 ***
## horsepower     3.483e+02  2.557e+01  13.622  < 2e-16 ***
## car_make       1.159e+05  6.369e+03  18.194  < 2e-16 ***
##
## Residual standard error: 55660 on 586 degrees of freedom
## Multiple R-squared:  0.6515, Adjusted R-squared:  0.6491
## F-statistic: 273.8 on 4 and 586 DF,  p-value: < 2.2e-16
```

```
mplot(MLR5, which = 1)
```

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



```
mplot(MLR5, which = 2)
```




```
set.seed(40)
randomizedsamples1 <- do(10000) * (anova(lm(shuffle(logprice) ~ car_make, data = SportsCars))$"F value"
randomizedsamples1 <- as.data.frame(randomizedsamples1)
```

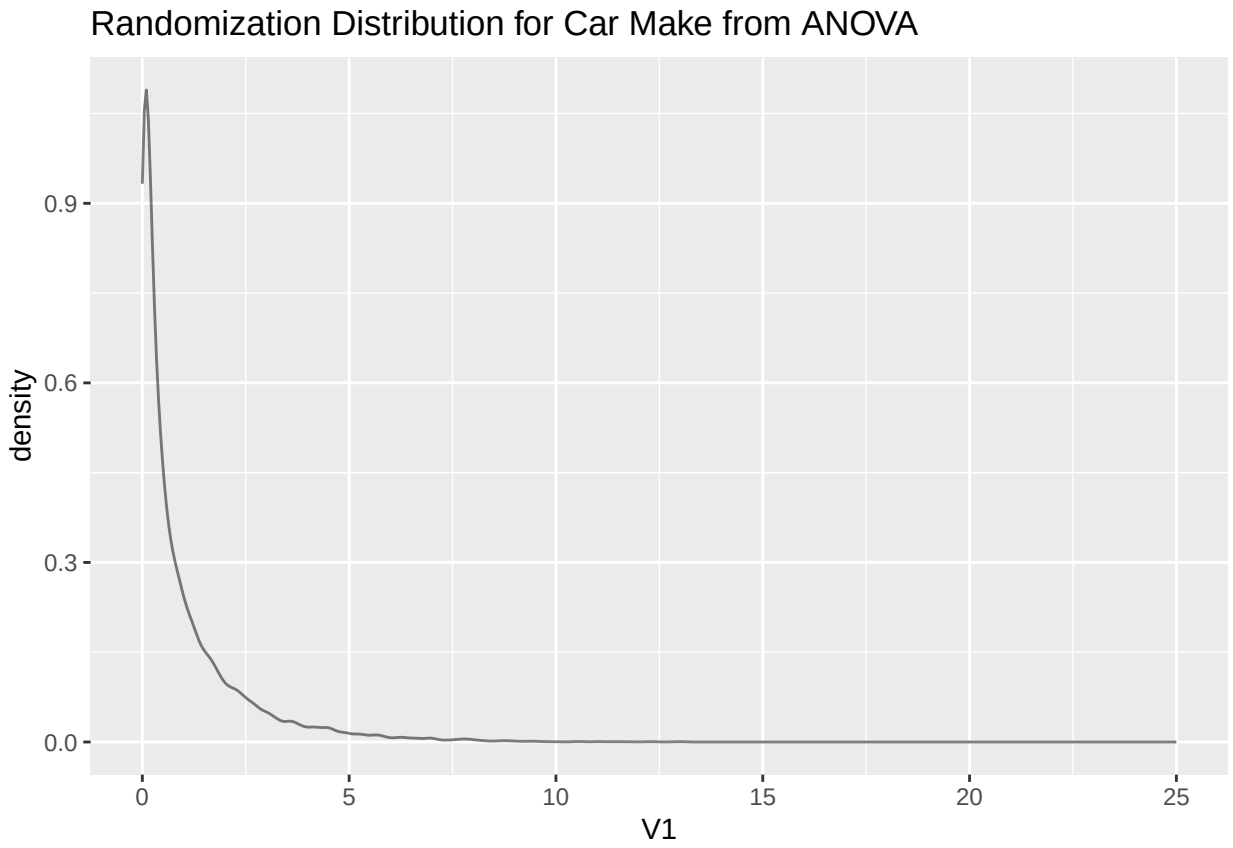
```
anova(loglocation)
```

```
## Analysis of Variance Table
##
## Response: logprice
##            Df  Sum Sq Mean Sq F value    Pr(>F)
## car_make    2  83.895  41.948  141.02 < 2.2e-16 ***
## Residuals 588 174.905   0.297
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
origres1 <- anova(loglocation)$"F value"#
origres1
```

```
## [1] 141.0201       NA
```

```
gf_dens(~ V1, data = randomizedsamples1) %>%
  gf_lims(x = c(0, 25)) %>%
  gf_labs(title = "Randomization Distribution for Car Make from ANOVA")
```




```
pdata(~ V1, origres[1], data = randomizedsamples1, lower.tail = FALSE)
```

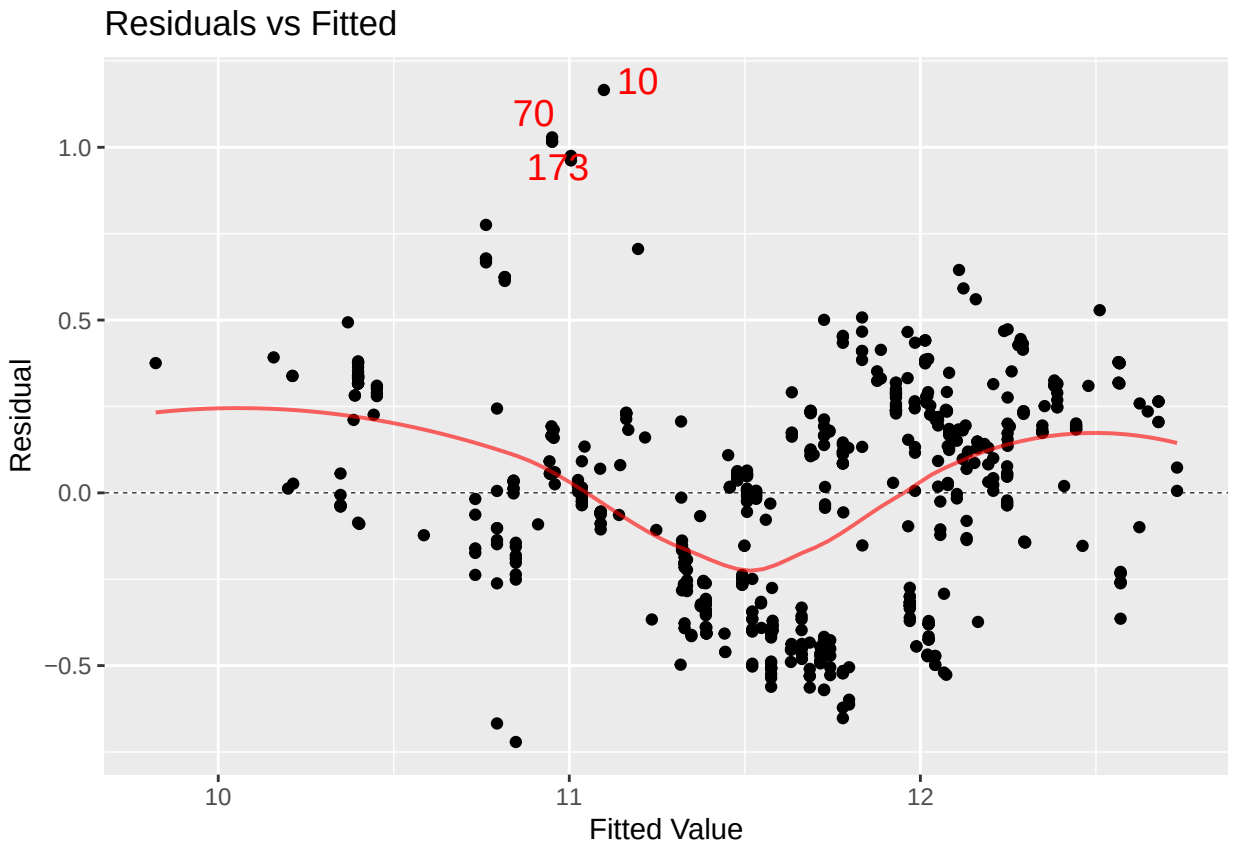
```
## [1] 0
```

```
MLR6<- lm(logprice ~ year + engine_size_l + horsepower + car_make, data=SportsCars)
msummary(MLR6)
```

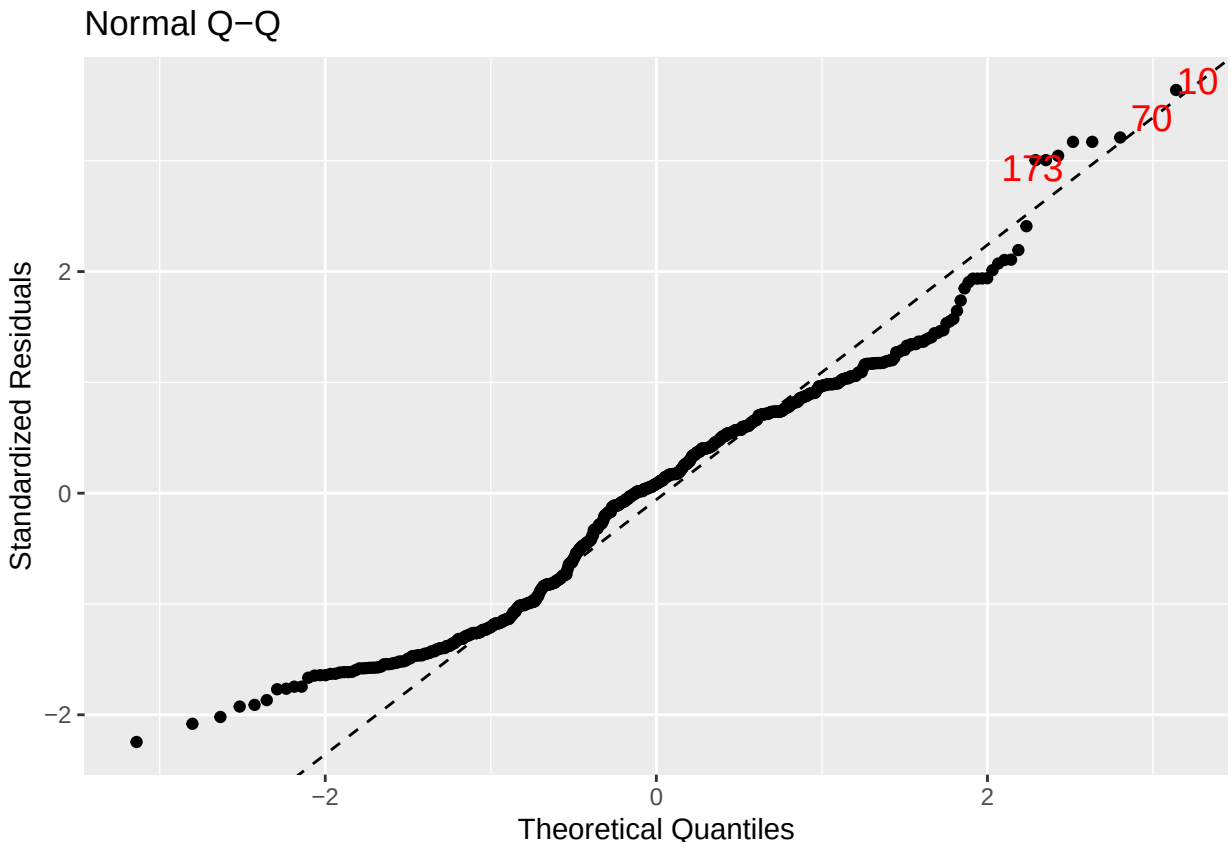
```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.176e+02  2.791e+01   4.212 2.93e-05 ***
## year          -5.362e-02  1.380e-02  -3.885 0.000114 ***
## engine_size_l  6.318e-02  1.650e-02   3.829 0.000143 ***
## horsepower     2.776e-03  1.485e-04  18.692  < 2e-16 ***
## car_make       9.374e-01  3.698e-02  25.346  < 2e-16 ***
##
## Residual standard error: 0.3232 on 586 degrees of freedom
## Multiple R-squared:  0.7634, Adjusted R-squared:  0.7618
## F-statistic: 472.7 on 4 and 586 DF,  p-value: < 2.2e-16
```

```
mplot(MLR6, which = 1)
```

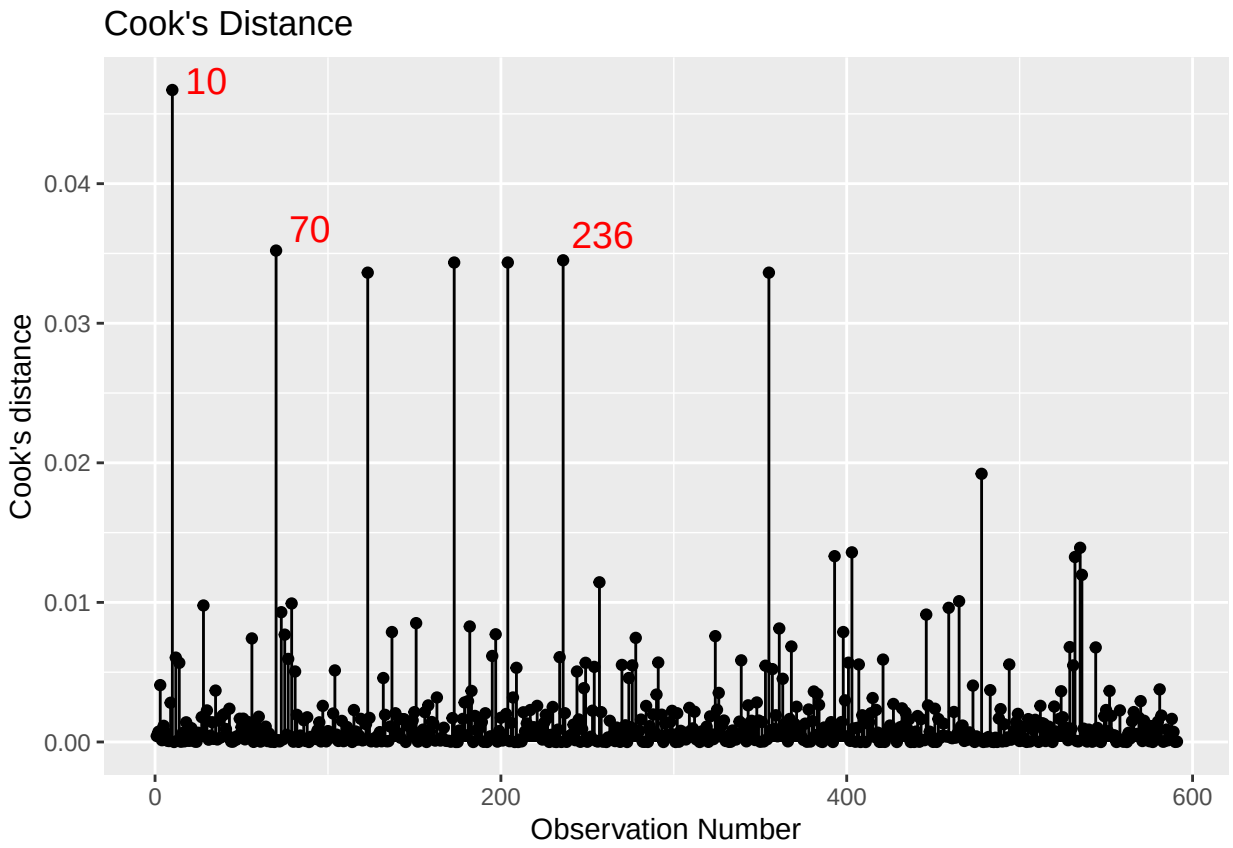
```
## `geom_smooth()` using formula = 'y ~ x'
```




```
mplot(MLR6, which = 2)
```




```
mplot(MLR6, which = 4)
```


Cook's Distance
10
70
236
0.04
0.03
0.02
0.01
0.00
Cook's distance
0
200
400
600
Observation Number