Building-Stat-Model.R

r1637457

2023-01-30

```
############################################################
############################################################

## Building Statistical Models in R: Linear Regression

############################################################
############################################################

############################################################
## Task One: Getting Started
## In this task, you will learn change the panes and font size.
## Also, you will learn how to set and check your current
## working directory
############################################################

## 1.1: Get the working directory
getwd()
```

```
## [1] "/cloud/project"
```

```
############################################################
## Task Two: Import packages and dataset
## In this task, you will import the required packages and data
## for this project
############################################################

## 2.1: Importing required packages
library(tidyverse)
library(ggpubr)
library(broom)
library(ggfortify)

## 2.2: Import the mpg.csv dataset
data <- read.csv("mpg.csv", header = T, sep = ",")

## 2.3: View and check the dimension of the dataset
view(data)

dim(data)
```

```
## [1] 234  12
```

```
############################################################
## Task Three: Explore the dataset
```

```
## In this task, you will learn how to explore and clean the data
############################################################

## 3.1: Take a peek using the head and tail functions
head(data)
```

```
##   X manufacturer model displ year cyl      trans drv cty hwy fl
## 1 1         audi    a4   1.8 1999   4   auto(l5)   f  18  29  p
## 2 2         audi    a4   1.8 1999   4 manual(m5)   f  21  29  p
## 3 3         audi    a4   2.0 2008   4 manual(m6)   f  20  31  p
## 4 4         audi    a4   2.0 2008   4   auto(av)   f  21  30  p
## 5 5         audi    a4   2.8 1999   6   auto(l5)   f  16  26  p
## 6 6         audi    a4   2.8 1999   6 manual(m5)   f  18  26  p
##     class
## 1 compact
## 2 compact
## 3 compact
## 4 compact
## 5 compact
## 6 compact
```

```
tail(data)
```

```
##       X manufacturer  model displ year cyl      trans drv cty
## 229 229   volkswagen passat   1.8 1999   4   auto(l5)   f  18
## 230 230   volkswagen passat   2.0 2008   4   auto(s6)   f  19
## 231 231   volkswagen passat   2.0 2008   4 manual(m6)   f  21
## 232 232   volkswagen passat   2.8 1999   6   auto(l5)   f  16
## 233 233   volkswagen passat   2.8 1999   6 manual(m5)   f  18
## 234 234   volkswagen passat   3.6 2008   6   auto(s6)   f  17
##     hwy fl   class
## 229  29  p midsize
## 230  28  p midsize
## 231  29  p midsize
## 232  26  p midsize
## 233  26  p midsize
## 234  26  p midsize
```

```
## 3.2: Check the internal structure of the data frame
str(data)
```

```
## 'data.frame':    234 obs. of  12 variables:
##  $ X           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
##  $ model       : chr  "a4" "a4" "a4" "a4" ...
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr  "f" "f" "f" "f" ...
##  $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr  "p" "p" "p" "p" ...
##  $ class       : chr  "compact" "compact" "compact" "compact" ...
```

```
## 3.3: Count missing values in the variables
sum(is.na(data))
```

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

```
sapply(data, function(x) sum(is.na(x)))
```

```
##            X manufacturer        model        displ         year
##            0            0            0            0            0
##          cyl        trans          drv          cty          hwy
##            0            0            0            0            0
##           fl        class
##            0            0
```

```
## 3.4: Check the column names for the data frame
names(data)
```

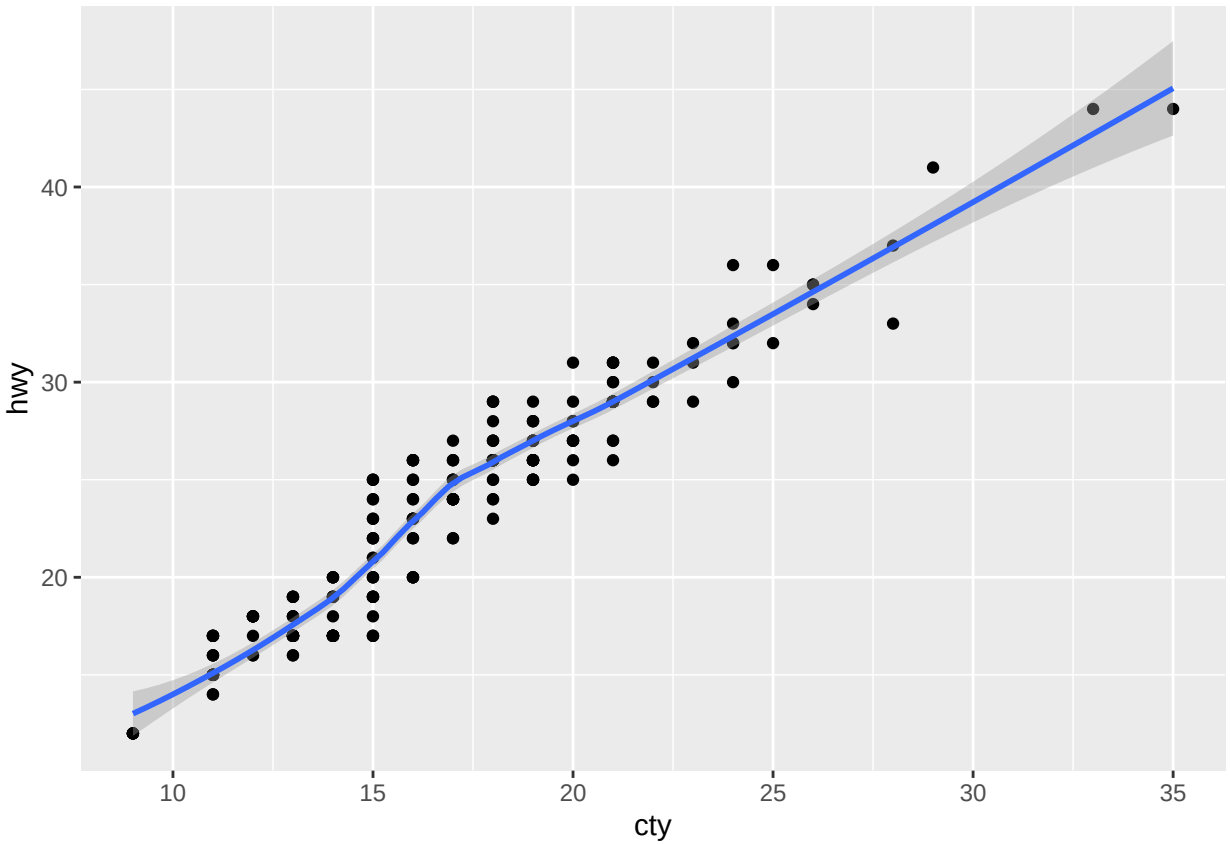
```
##  [1] "X"            "manufacturer" "model"        "displ"
##  [5] "year"         "cyl"          "trans"        "drv"
##  [9] "cty"          "hwy"          "fl"           "class"
```

```
## 3.5: Drop the first column of the data frame
data <-data[, -1]

dim(data)
```

```
## [1] 234  11
```

```
################################################################
## Task Four: Data Visualizations
## In this task, you will learn how to visualize the variables
## we will use to build the statistical model
################################################################

## 4.1: Plot a scatter plot for the variables with cty on the x-axis
## hwy on the y-axis
ggplot(data, aes(x = cty, y= hwy)) +
  geom_point() +
  stat_smooth()
```

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




```
## 4.2: Find the correlation between the variables
cor(data$cty, data$hwy)
```

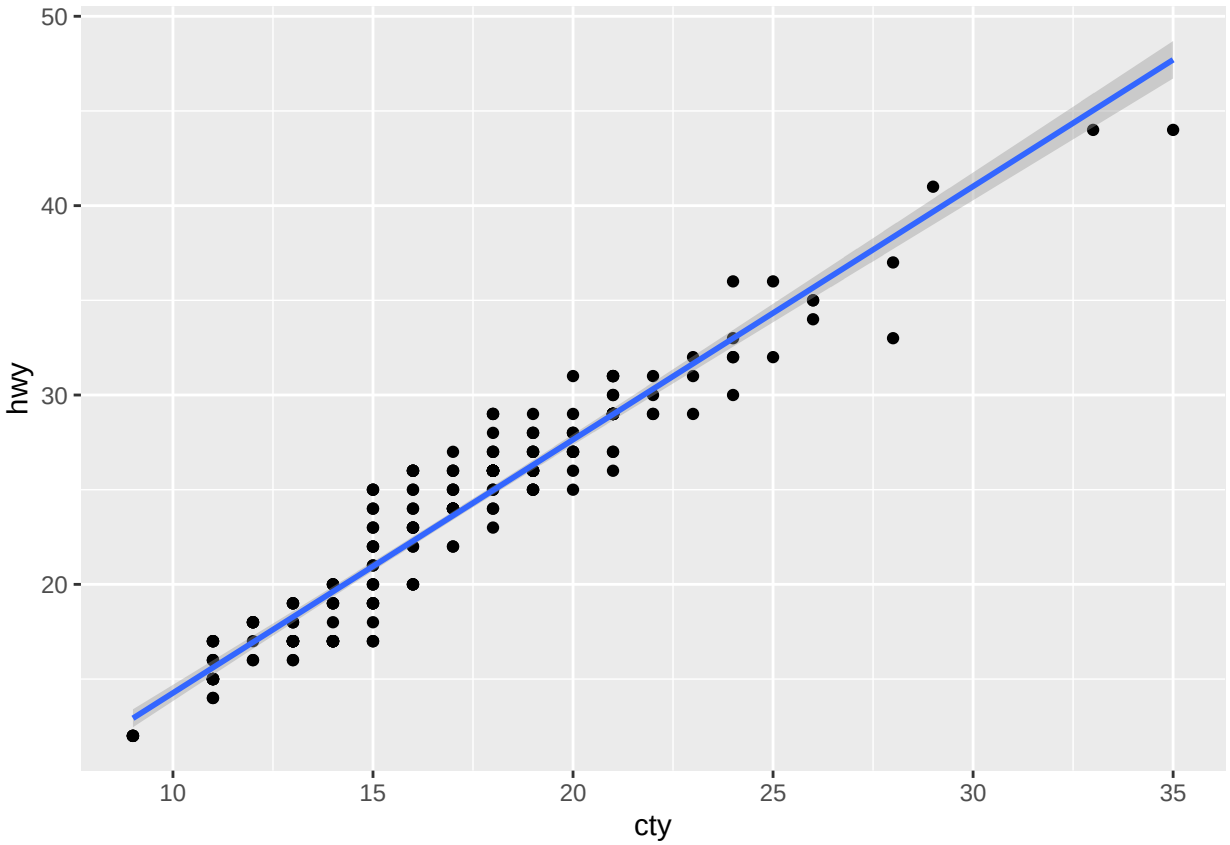
```
## [1] 0.9559159
```

```
#################################################################
## Task Five: Model Building
## In this task, you will learn how to build a simple
## linear regression model
#################################################################

## 5.1: Create a simple linear regression model using the variables
model <- lm(hwy ~ cty, data = data)
model
```

```
##
## Call:
## lm(formula = hwy ~ cty, data = data)
##
## Coefficients:
## (Intercept)          cty
##       0.892        1.337
```

```
## 5.2: Plot the regression line for the model
ggplot(data, aes(x = cty, y= hwy)) +
  geom_point() +
  stat_smooth(method = lm)
```

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




```
############################################################
## Task Six: Model Assessment I
## In this task, you will learn how to assess and interpret
## the result of a simple linear regression model
############################################################

## 6.1: Assess the summary of the fitted model
summary(model)
```

```
##
## Call:
## lm(formula = hwy ~ cty, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3408 -1.2790  0.0214  1.0338  4.0461
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.89204    0.46895   1.902   0.0584 .
## cty          1.33746    0.02697  49.585   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.752 on 232 degrees of freedom
## Multiple R-squared:  0.9138, Adjusted R-squared:  0.9134
## F-statistic:  2459 on 1 and 232 DF,  p-value: < 2.2e-16
```

```
## 6.2: Calculate the confidence interval for the coefficients
confint(model)
```

```
##                   2.5 %   97.5 %
## (Intercept) -0.03189534 1.815978
## cty          1.28431197 1.390599
```

```
#################################################################
## Task Seven: Model Assessment II
## In this task, you will learn how to assess the accuracy
## of a simple linear regression model
#################################################################

## 7.1: Assess the summary of the fitted model
summary(model)
```

```
##
## Call:
## lm(formula = hwy ~ cty, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.3408 -1.2790  0.0214  1.0338  4.0461
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.89204    0.46895   1.902   0.0584 .
## cty          1.33746    0.02697  49.585   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.752 on 232 degrees of freedom
## Multiple R-squared:  0.9138, Adjusted R-squared:  0.9134
## F-statistic:  2459 on 1 and 232 DF,  p-value: < 2.2e-16
```

```
## 7.2: Calculate the prediction error of the fitted model
sigma(model)*100/mean(data$hwy)
```

```
## [1] 7.475581
```

```
#################################################################
## Task Eight: Model Prediction
## In this task, you will learn how to check for metrics from
## the fitted model and make prediction for new values
#################################################################

## 8.1: Find the fitted values of the simple regression model
fitted <- predict.lm(model)
head(fitted, 3)
```

```
##        1        2        3
## 24.96624 28.97861 27.64115
```

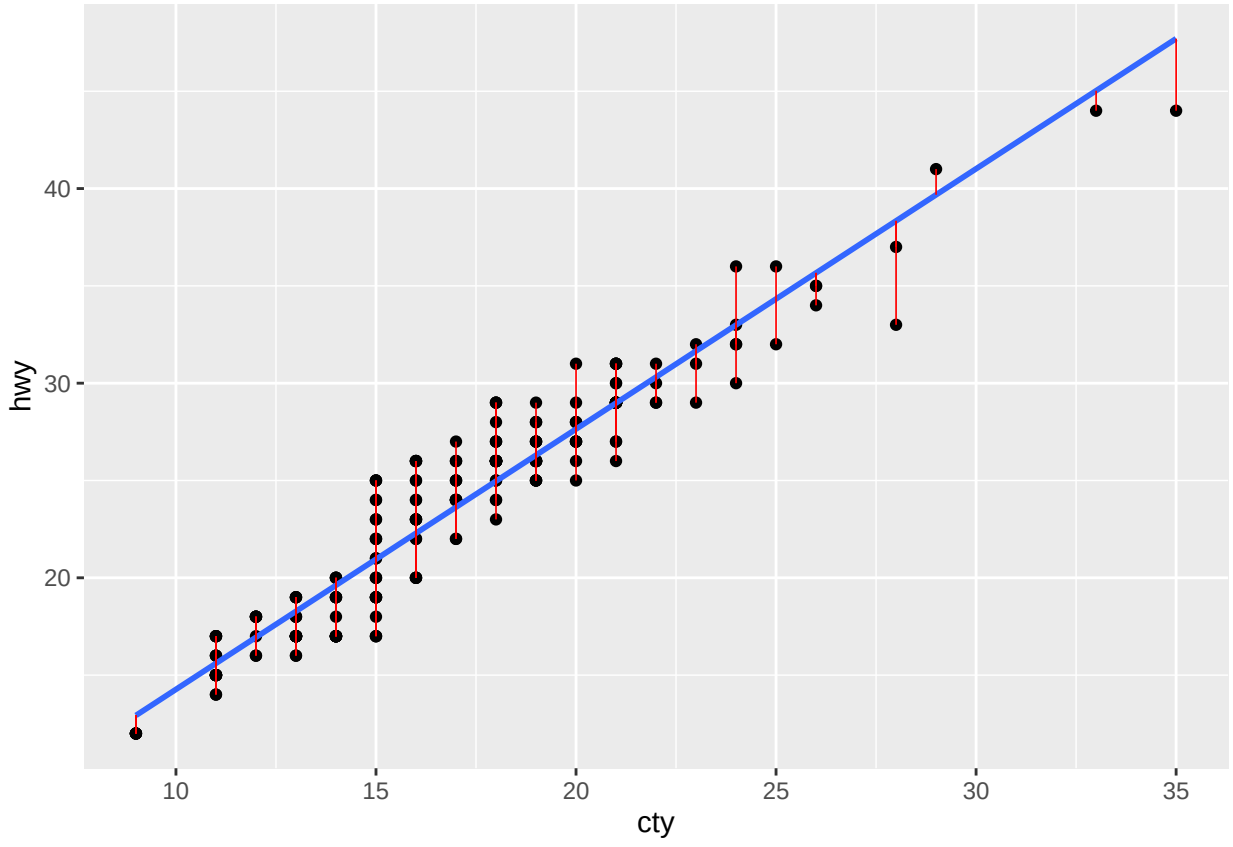
```
## 8.2: Find the fitted values of the simple regression model
model_diag_metrics <- augment(model)
head(model_diag_metrics)
```

```
## # A tibble: 6 x 8
##     hwy   cty .fitted .resid    .hat .sigma     .cooksd .std.re~1
##   <int> <int>   <dbl>  <dbl>   <dbl>  <dbl>       <dbl>     <dbl>
## 1    29    18    25.0 4.03   0.00458   1.74 0.0123          2.31
## 2    29    21    29.0 0.0214 0.00834   1.76 0.000000632     0.0123
## 3    31    20    27.6 3.36   0.00661   1.74 0.0123          1.92
## 4    30    21    29.0 1.02   0.00834   1.75 0.00144         0.585
## 5    26    16    22.3 3.71   0.00445   1.74 0.0101          2.12
## 6    26    18    25.0 1.03   0.00458   1.75 0.000805        0.591
## # ... with abbreviated variable name 1: .std.resid
```

```
## 8.3: Visualize the residuals of the fitted model
ggplot(model_diag_metrics, aes(cty, hwy)) +
  geom_point() +
  stat_smooth(method = lm, se = FALSE) +
  geom_segment(aes(xend = cty, yend = .fitted), color = "red", size = 0.3)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2
## 3.4.0.
## i Please use `linewidth` instead.
```

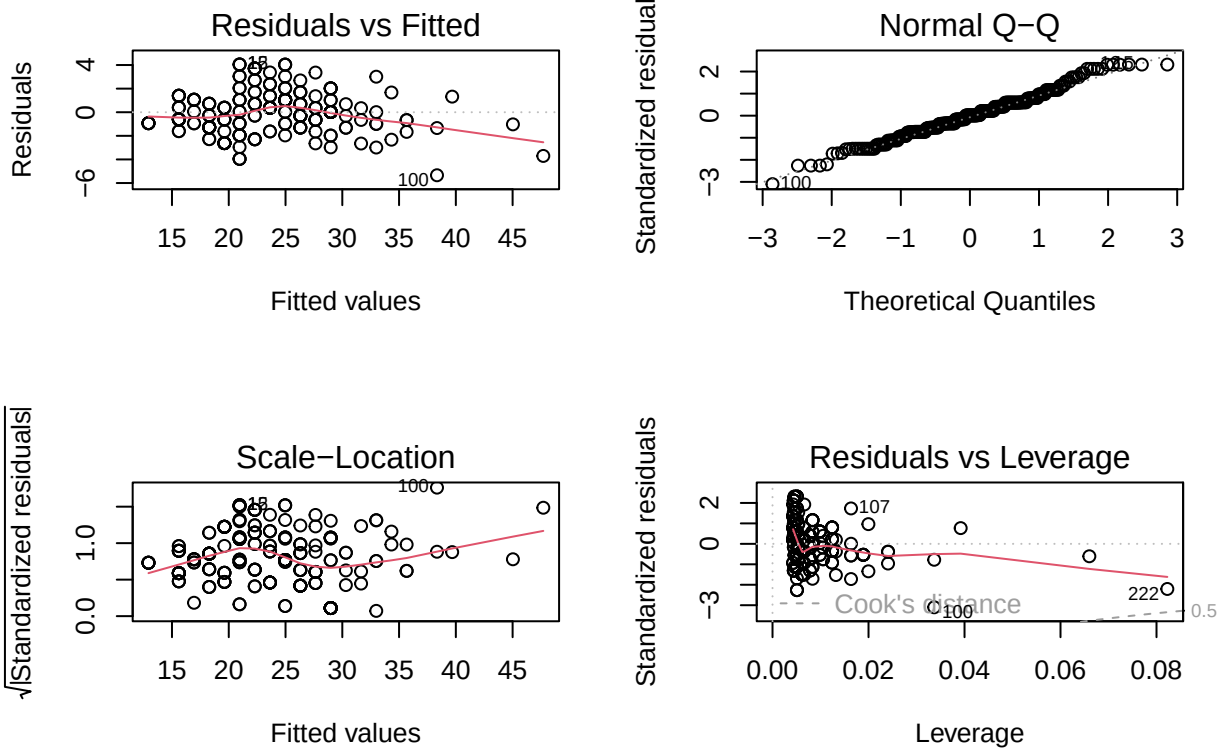
```
## `geom_smooth()` using formula = 'y ~ x'
```



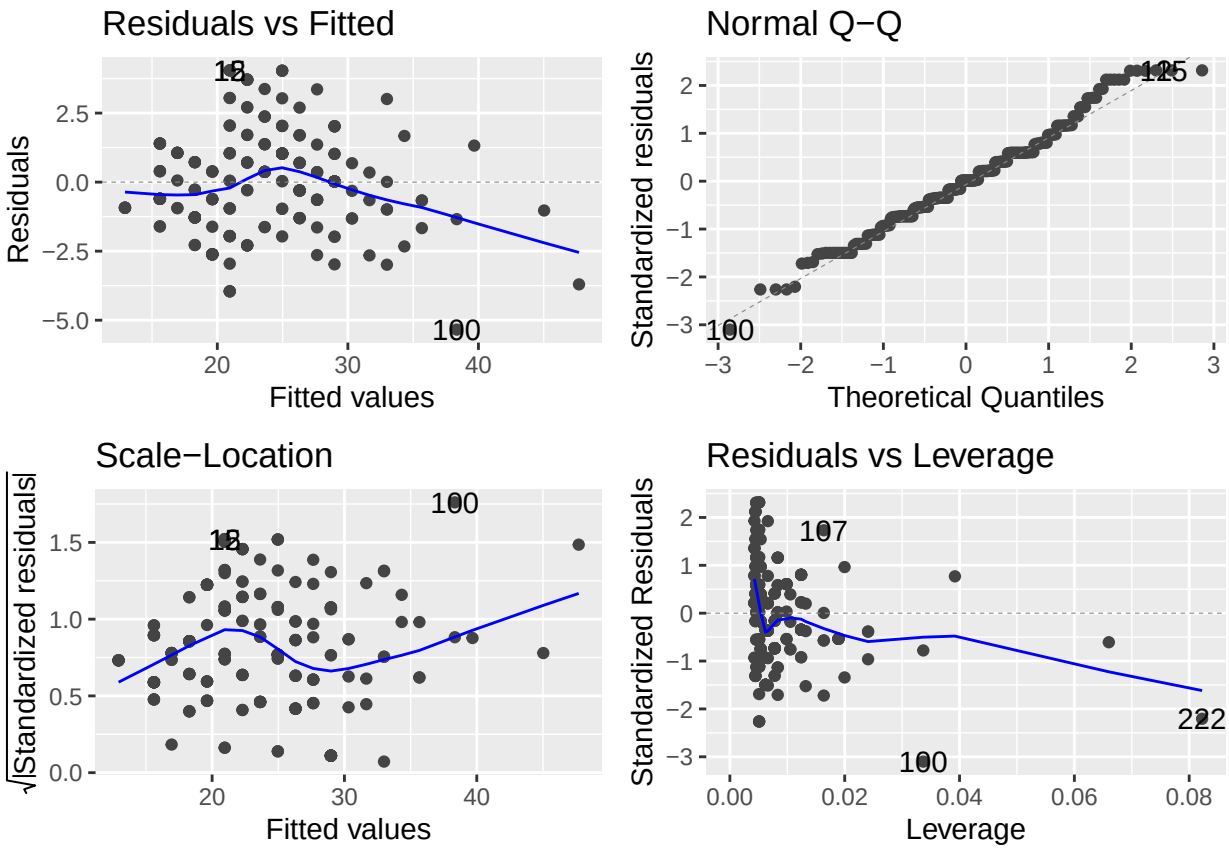

```
## 8.4: Predict new values using the model
predict(
  object = model,
  newdata = data.frame(cty = c(21,27,14))
)
```

```
##        1        2        3
## 28.97861 37.00334 19.61642
```

```
############################################################
## Task Nine: Assumptions Check: Diagnostic Plots
## In this task, you will learn how to perform diagnostics
## check on the fitted model
############################################################

## 9.1: Plotting the fitted model
par(mfrow = c(2, 2))   ## This plots the figures in a 2 x 2
plot(model)
```



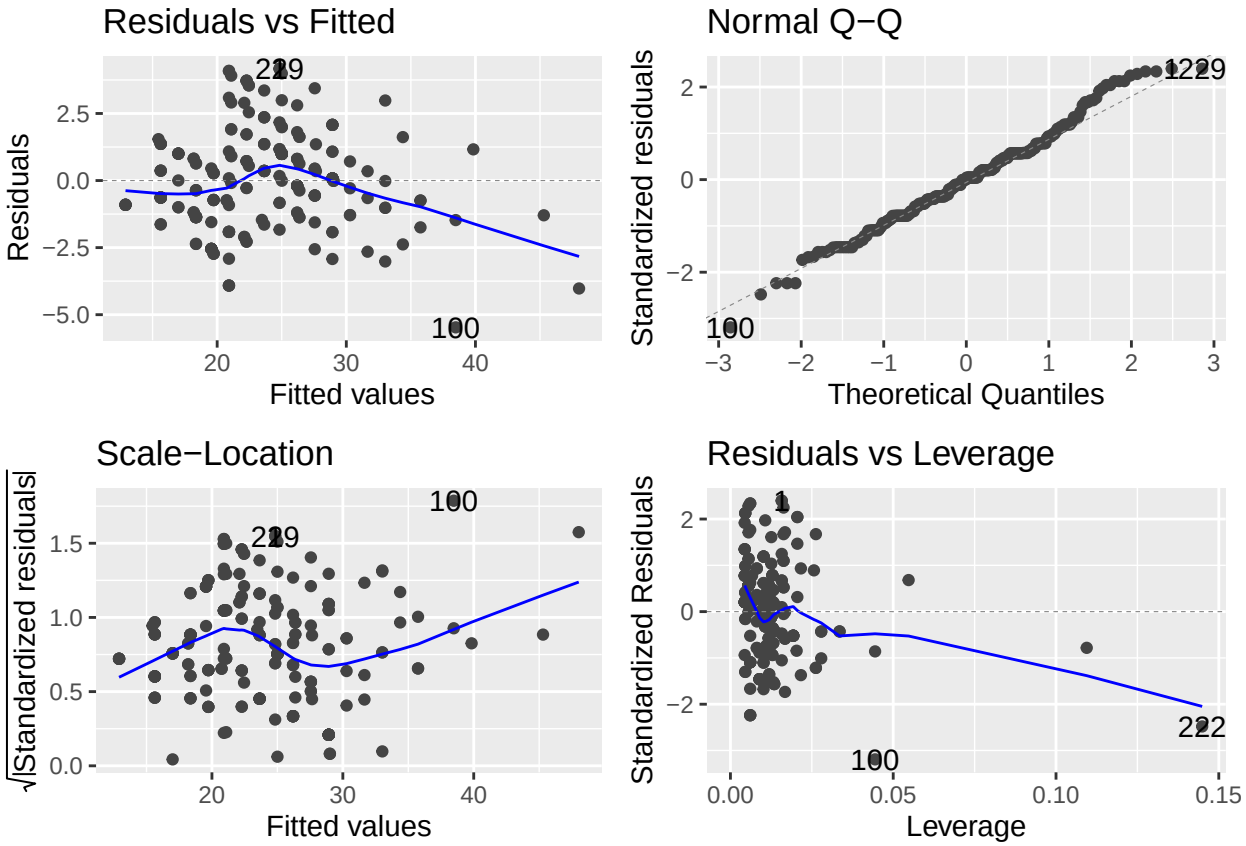

```
## Better Version
autoplot(model)
```



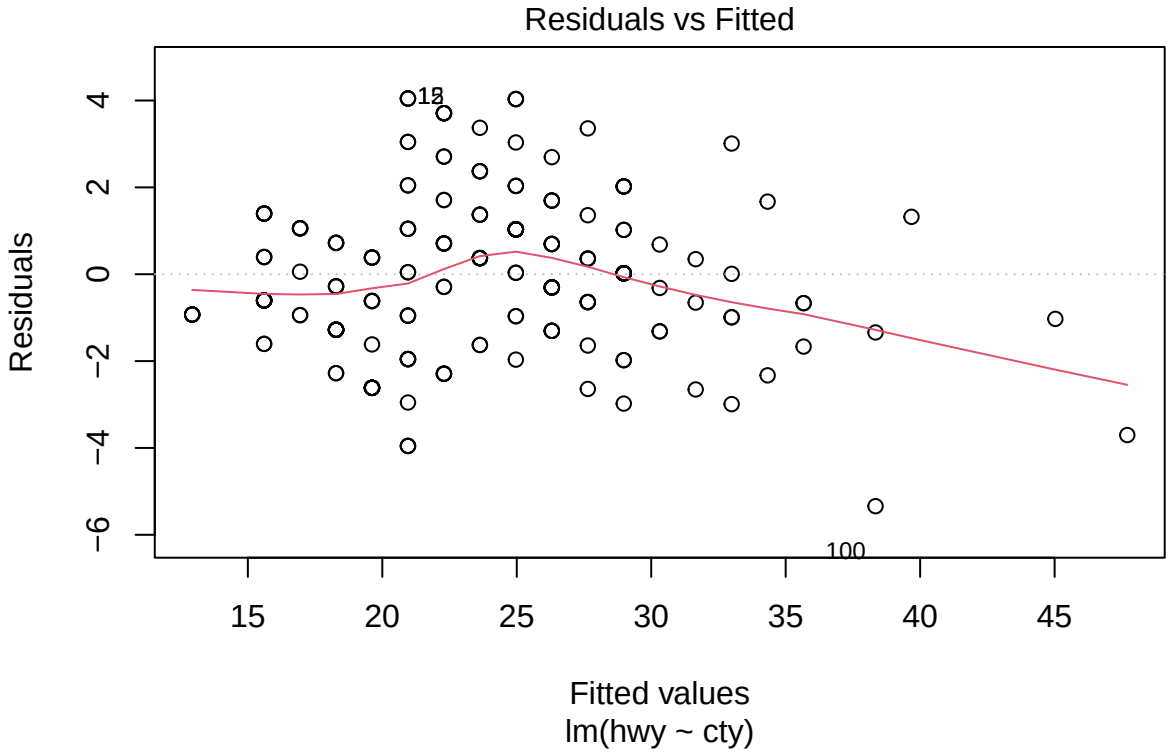

```
## 9.2: Return par back to default
dev.off()
```

```
## RStudioGD
##         2
```

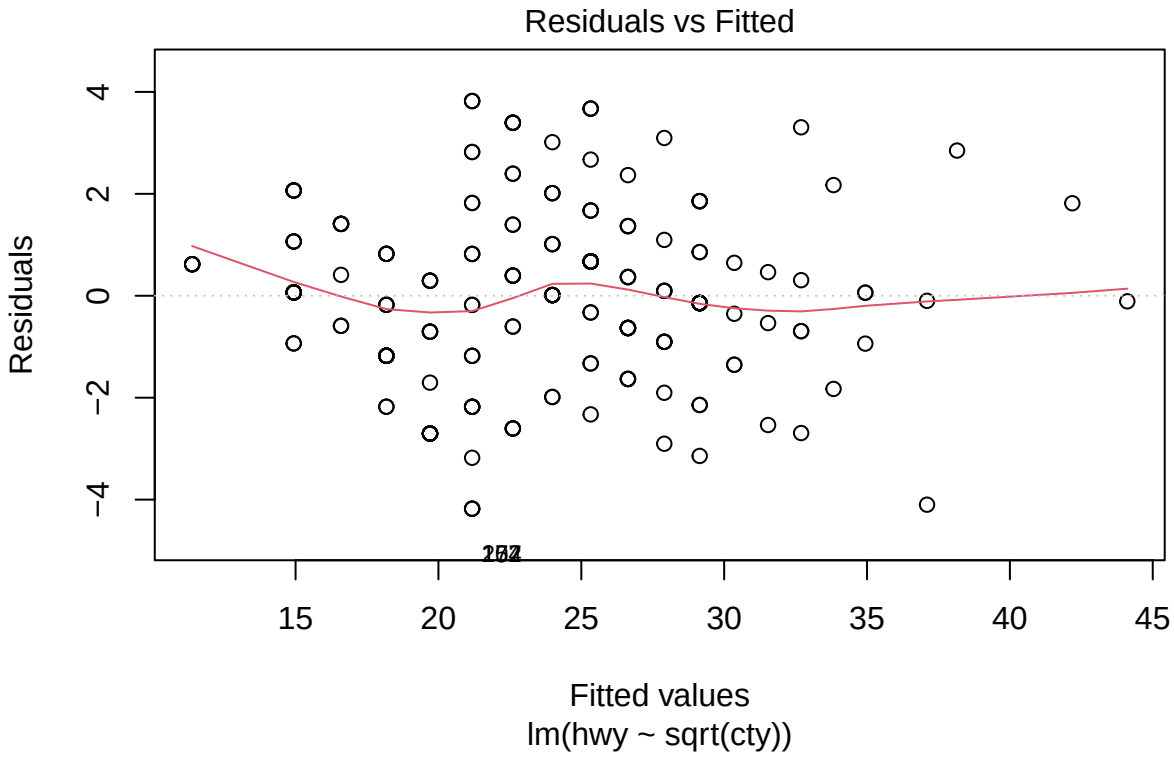
```
## or
par(mfrow = c(1, 1))
```



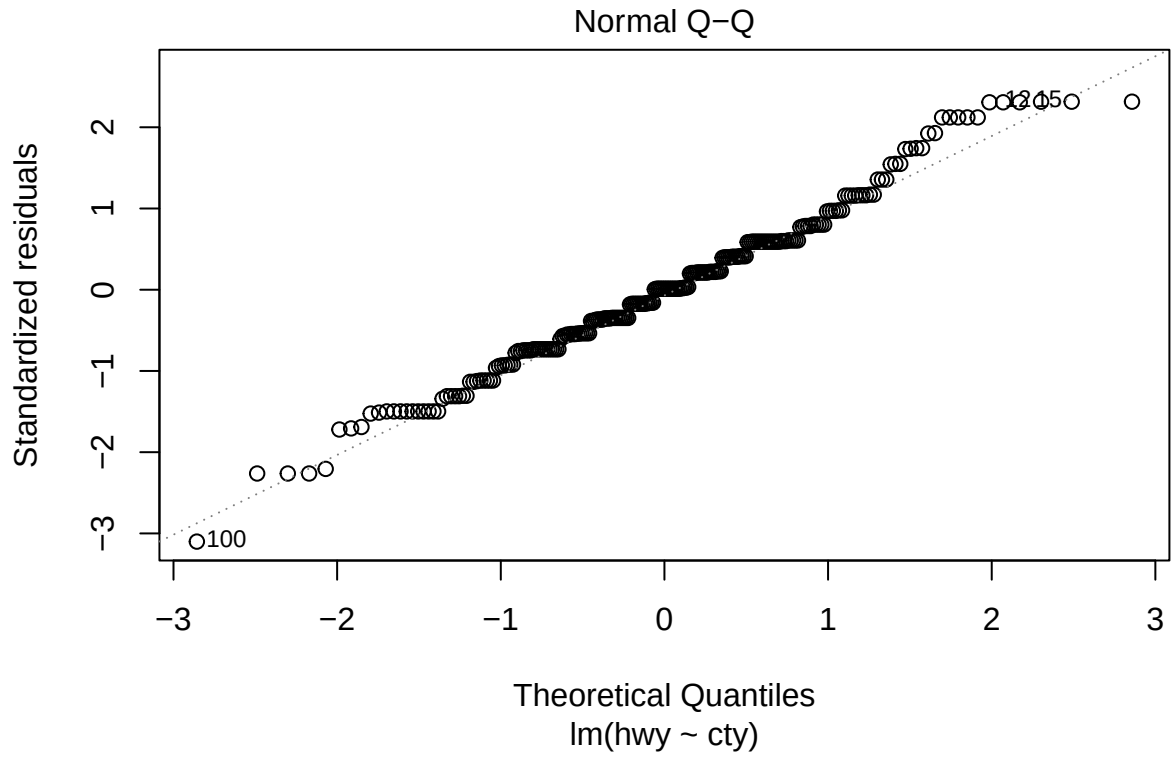

```
## 9.3: Return the first diagnostic plot for the model
plot(model,1)
```



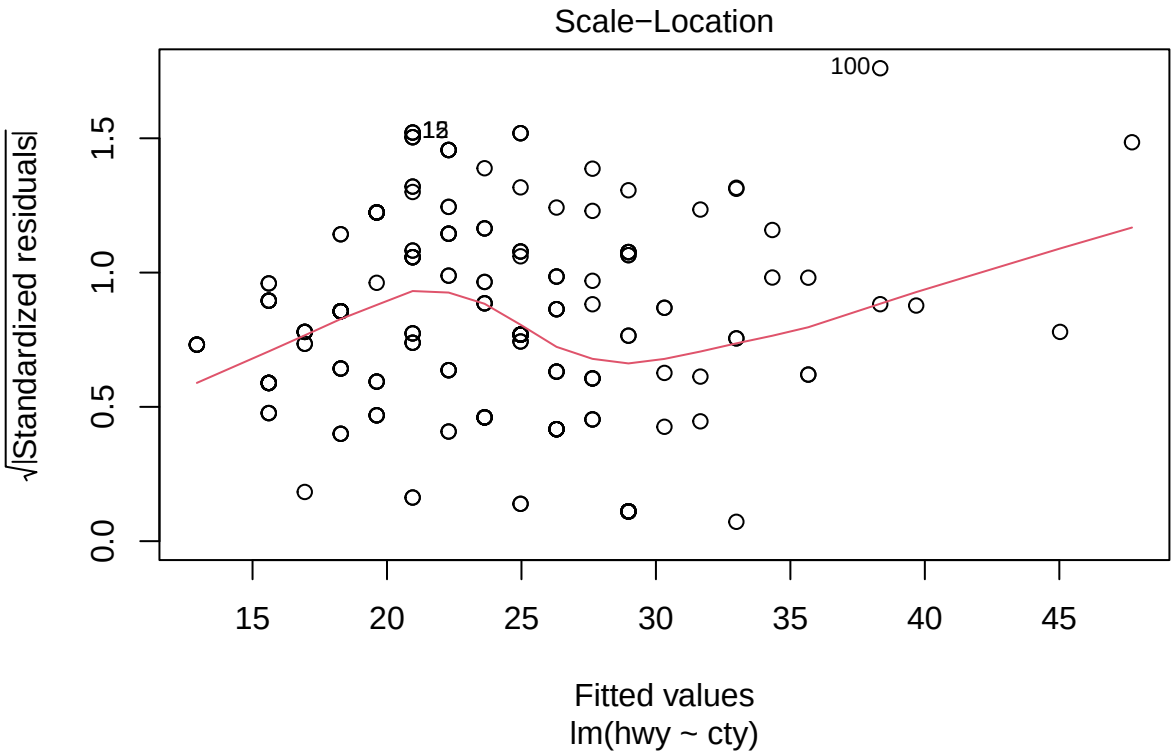
```r
## Build another regression model
model1 <- lm(hwy ~ sqrt(cty), data = data)
plot(model1, 1)
```




```r
## 9.4: Return the second diagnostic plot for the model
plot(model,2)
```




```
## 9.5: Return the third diagnostic plot for the model
plot(model,3)
```




```
############################################################
## Task Ten: Multiple Regression
## In this task, you will learn how to build and interpret the results
```

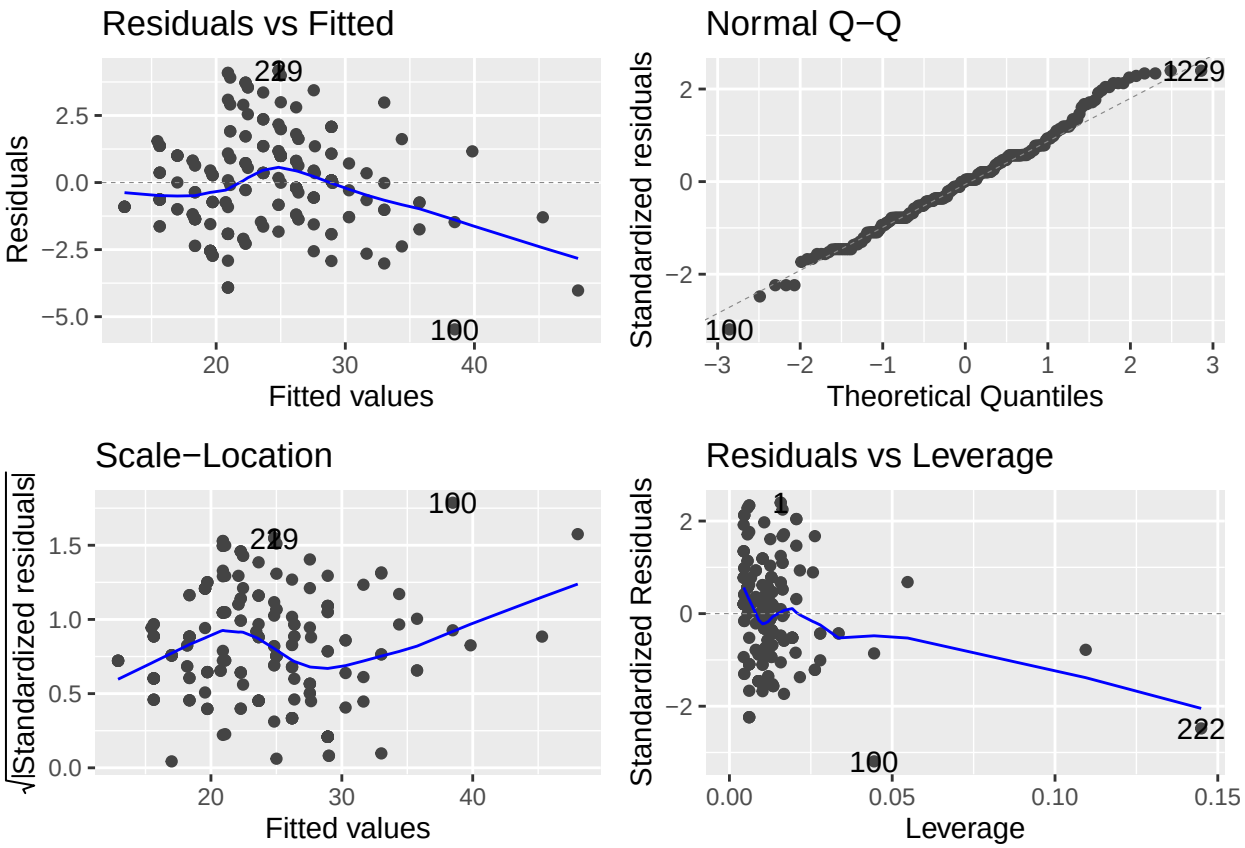
```r
## of a multiple regression model
############################################################

## 10.1: Build the multiple regression model with hwy on the y-axis and
## cty and cyl on the x-axis
mul_reg_model <- lm(hwy ~ cty + cyl, data = data)

## 10.2: This prints the result of the model
mul_reg_model
```

```
##
## Call:
## lm(formula = hwy ~ cty + cyl, data = data)
##
## Coefficients:
## (Intercept)          cty          cyl
##    -0.07702      1.36425      0.08784
```

```r
## 10.3: Check the summary of the multiple regression model
summary(mul_reg_model)
```

```
##
## Call:
## lm(formula = hwy ~ cty + cyl, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4735 -1.1952  0.0398  0.9934  4.1691
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.07702    1.40888  -0.055    0.956
## cty          1.36425    0.04559  29.924   <2e-16 ***
## cyl          0.08784    0.12040   0.730    0.466
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.754 on 231 degrees of freedom
## Multiple R-squared:  0.914,  Adjusted R-squared:  0.9132
## F-statistic:  1227 on 2 and 231 DF,  p-value: < 2.2e-16
```

```r
## 10.4: Plot the fitted multiple regression model
autoplot(mul_reg_model)
```


Residuals vs Fitted
Residuals
Fitted values
Normal Q−Q
Standardized residuals
Theoretical Quantiles
Scale−Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized Residuals
Leverage