HW11

106022103

2021/5/8

Assist

- 106000199
  - Helped me how to get factor names in data.frame.
  - Disscussed about the Q2.a, which visualization should take.

Set up

import libary

```
library(ggplot2)
require(qqplotr)
library(plyr)
library(gridExtra)
library(ggcorrplot)
library(magrittr)
library(ggpubr)
```

Q1

(a) Let's dig into what regression is doing to compute model fit

Because `interactive_regression` can't run in Rmarkdown knit, we have to run these commands in console and save the variables `pts`.

```
pts <- interactive_regression()
saveRDS(pts, file = "W:/Rtmp/pts.rds")
```

```
pts <- readRDS(file = "W:/Rtmp/pts1.rds")
```

i. Plot Scenario 2, storing the returned points: `pts <- interactive_regression_rsq()`

```
regr <- lm(y ~ x, data=pts)
summary(regr)
```

ii. Run a linear model of x and y points to confirm the R2 value reported by the simulation:

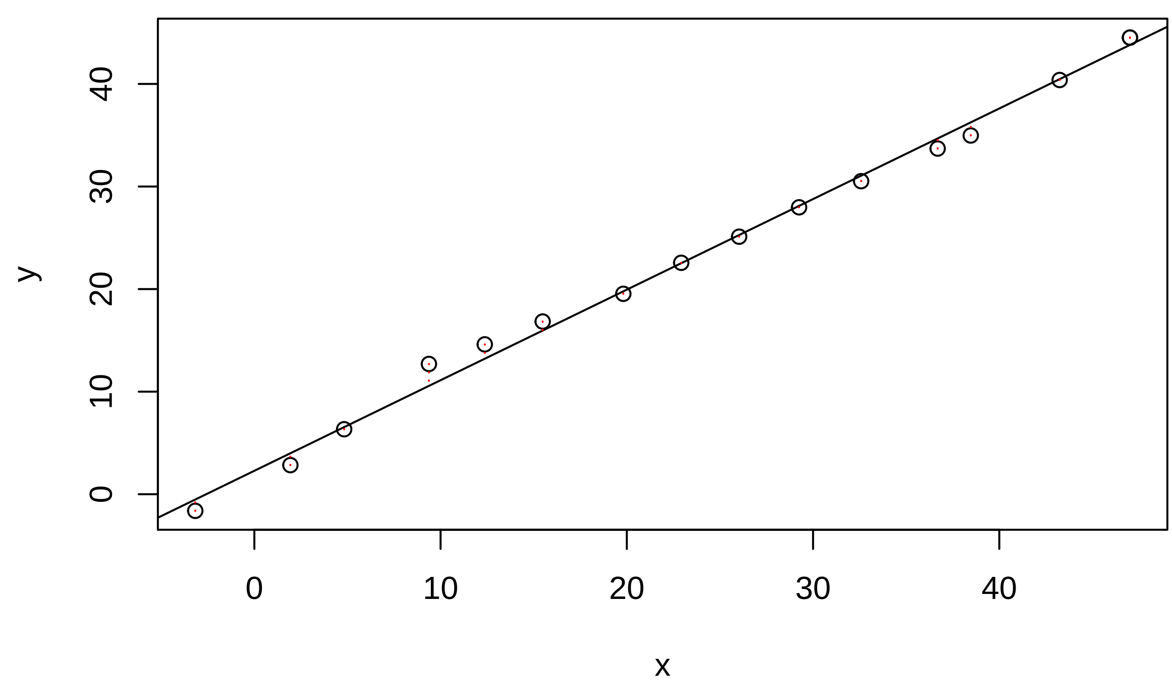
```
##
## Call:
## lm(formula = y ~ x, data = pts)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.2791 -0.6484 -0.1510  0.7166  2.1375
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.28297    0.45554   5.012  0.00019 ***
## x            0.88307    0.01589  55.569  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9973 on 14 degrees of freedom
## Multiple R-squared:  0.9955, Adjusted R-squared:  0.9952
## F-statistic:  3088 on 1 and 14 DF,  p-value: < 2.2e-16
```

iii. Add line segments to the plot to show the regression residuals (errors) as follows:

- Get values of y(regression line's estimates of y, given x): y_hat <- regr$fitted.values
- Add segments: segments(ptsx, ptsy, pts$x, y_hat, col="red", lty="dotted")

```
pts_regr <- lm(y~x, data=pts)
y_hat <- pts_regr$fitted.values

plot(pts)
abline(pts_regr)
segments(pts$x, pts$y, pts$x, y_hat, col="red", lty="dotted")
```



```r
SSE <- sum((pts$y-y_hat)^2)
SSR <- sum((y_hat-mean(pts$y))^2)
SST <-  SSE + SSR
R2 <- SSR/SST
cat(sprintf("SSE\tSSR\tSST\tR^2\n%.2f\t%.2f\t%.2f\t%.2f\n", SSE, SSR, SST, R2))
```

iv. Use only ptsx, ptsy, y_hat and mean(pts$y) to compute SSE, SSR and SST, and verify R2

```
## SSE  SSR SST R^2
## 13.93    3071.43 3085.35 1.00
```

(b) Comparing scenarios 1 and 2, which do we expect to have a stronger R^2 ?

ANSWER: scenarios 1.

(c) Comparing scenarios 3 and 4, which do we expect to have a stronger R^2 ?

ANSWER: scenarios 3.

(d) Comparing scenarios 1 and 2, which do we expect has bigger/smaller SSE, SSR, and SST?

(do not compute SSE/SSR/SST here – just provide your intuition)

ANSWER: scenarios 2.

(e) Comparing scenarios 3 and 4, which do we expect has bigger/smaller SSE, SSR, and SST?

(do not compute SSE/SSR/SST here – just provide your intuition)

ANSWER: scenarios 4.

Q2

Read File

```r
auto <- read.table("data/auto-data.txt", header=FALSE, na.strings = "?")
names(auto) <- c("mpg", "cylinders", "displacement", "horsepower", "weight",
                 "acceleration", "model_year", "origin", "car_name")
```

(a) data explore

i. Visualize the data in any way

```r
cylinder_freq <- as.data.frame(table(auto$cylinders))
origin_freq <- as.data.frame(table(auto$origin))
year_freq <- as.data.frame(table(auto$model_year))

names(cylinder_freq) <- c("cylinder","Freq")
names(origin_freq) <- c("origin","Freq")
names(year_freq) <- c("model_year","Freq")

sum(is.na(auto)) # How many na values
```

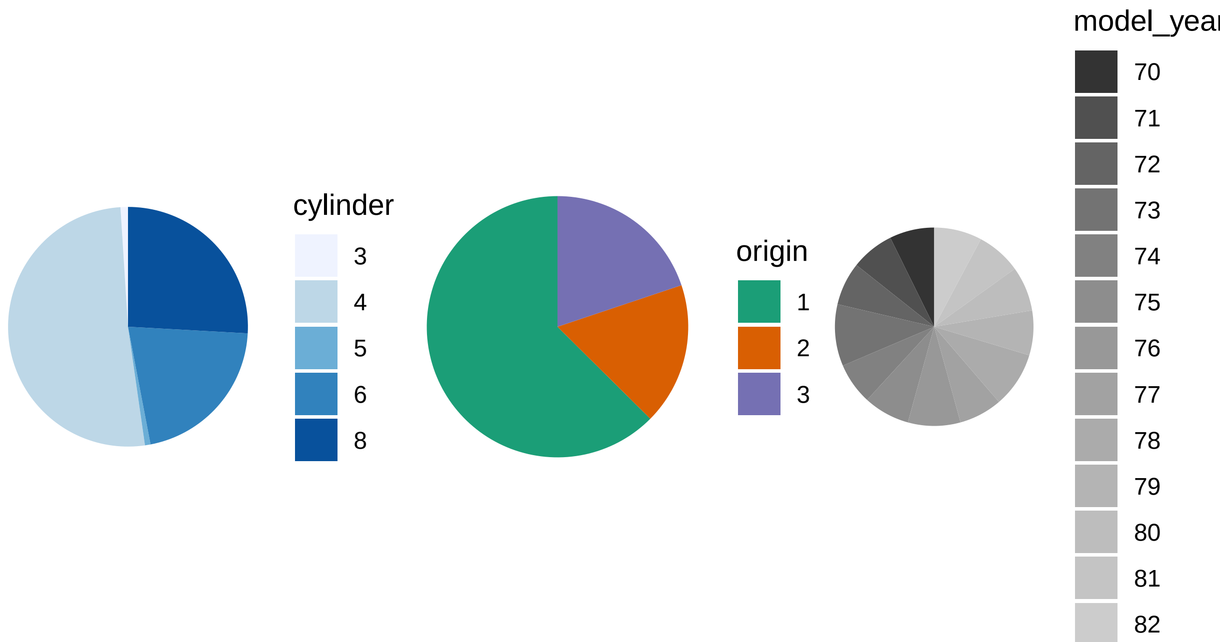
Preprocess

```
## [1] 6
```

```r
auto <- auto[complete.cases(auto), ] # remove missing value

auto[,'origin']<-factor(auto[,'origin']) # convert to factor
auto[,'car_name']<-factor(auto[,'car_name']) # convert to factor

auto_value <- auto[,-8:-9] # drop the class data
corr <- round(cor(auto_value), 2)
p.mat <- cor_pmat(auto_value)
```

Since there are no so many missing values, so I decided to just remove them.

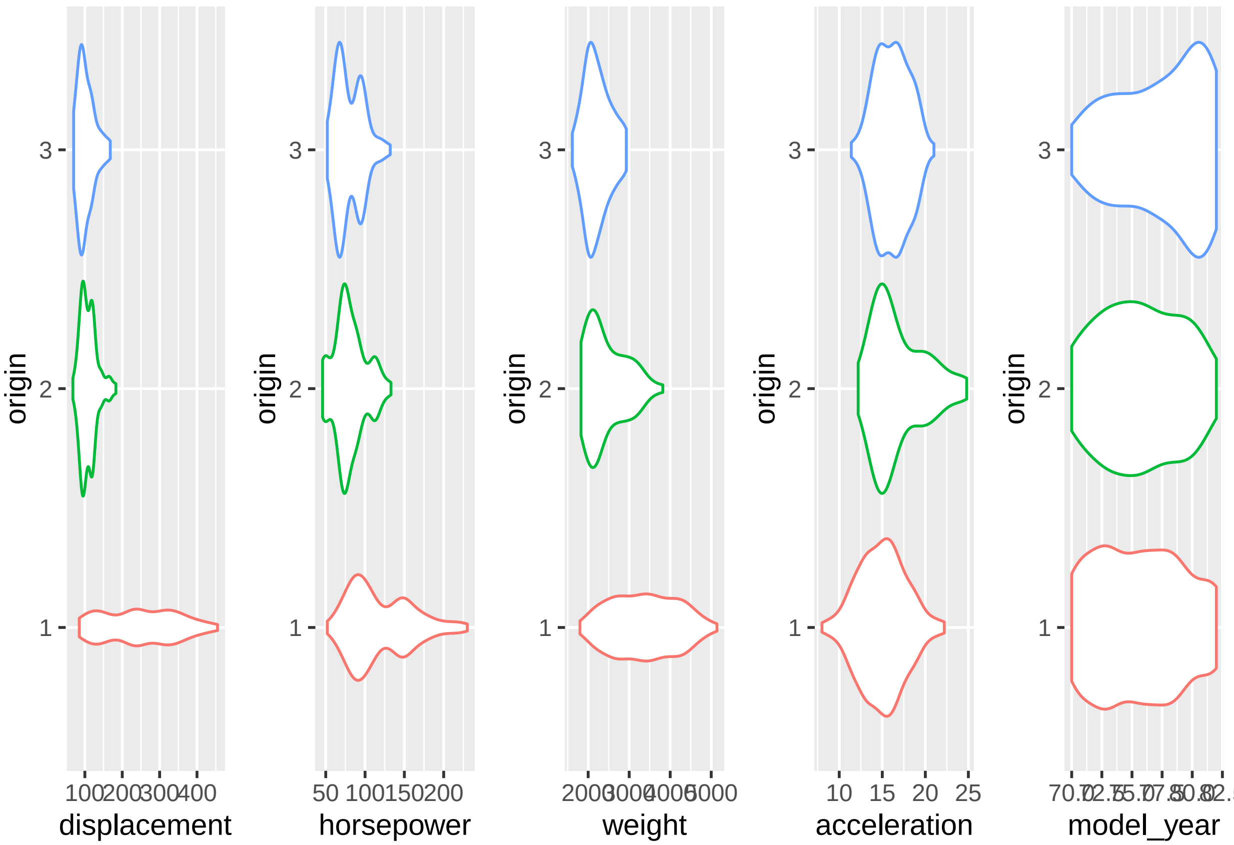
```r
p1 <- ggplot(data=cylinder_freq) +
    geom_bar(aes(x=factor(1),
                 y=Freq,
                 fill=cylinder),
             stat = "identity") +
    coord_polar("y", start=0) +
    scale_fill_brewer(palette="Blues") +
    theme_void() # remove background

p2 <- ggplot(data=origin_freq) +
    geom_bar(aes(x=factor(1),
                 y=Freq,
                 fill=origin),
             stat = "identity") +
    coord_polar("y", start=0) +
    scale_fill_brewer(palette="Dark2") +
    theme_void() # remove background

p3 <- ggplot(data=year_freq) +
    geom_bar(aes(x=factor(1),
                 y=Freq,
                 fill=model_year),
             stat = "identity") +
    coord_polar("y", start=0) +
    scale_fill_grey() +
    theme_void() # remove background

grid.arrange(p1,p2,p3,nrow=1,ncol=3)
```



Pie Chart

```
p4 <- ggplot(auto, aes(x=origin, y=displacement, color=origin)) +
  geom_violin() +
  coord_flip() +
  theme(legend.position="none")

p5 <- ggplot(auto, aes(x=origin, y=horsepower, color=origin)) +
  geom_violin() +
  coord_flip() +
  theme(legend.position="none")

p6 <- ggplot(auto, aes(x=origin, y=weight, color=origin)) +
  geom_violin() +
  coord_flip() +
  theme(legend.position="none")

p7 <- ggplot(auto, aes(x=origin, y=acceleration, color=origin)) +
  geom_violin() +
  coord_flip() +
  theme(legend.position="none")

p8 <- ggplot(auto, aes(x=origin, y=model_year, color=origin)) +
  geom_violin() +
  coord_flip() +
  theme(legend.position="none")
```

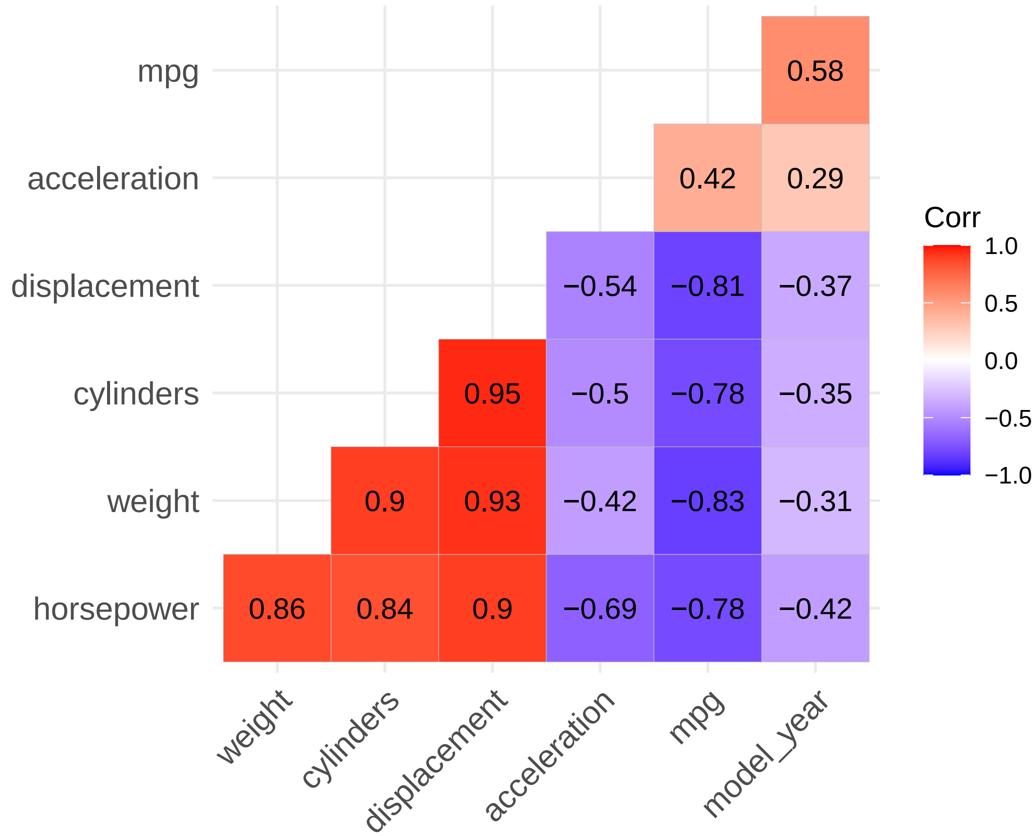
```
grid.arrange(p4,p5,p6,p7,p8, nrow=1,ncol=5)
```



Violin plot

ii. Report a correlation table of all variables Corr matrix

```
ggcorrplot(corr, hc.order = TRUE,
           type = "lower", p.mat = p.mat, lab = TRUE)
```



iii. which variables seem to relate to mpg? ANSWER: Take 0.7 as the threshold value, `mpg` is related to `displacement`, `cylinders`, `weight`, `horsepower`.

```
# ref.7
#    p-value   data frame
flattenCorrMatrix <- function(cormat) {
  ut <- upper.tri(cormat) # Lower and Upper Triangular Part of a Matrix
  data.frame(
    var1 = rownames(cormat)[row(cormat)[ut]],
    var2 = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut]
  )
}
cor_table <- flattenCorrMatrix(corr)
```

iv. Which relationships might not be linear? ANSWER: Take 0.5 as the threshold value, the following relationships may not be linear:

```
cor_table %>% dplyr::filter(abs(cor) < 0.5)
```

```
##           var1         var2   cor
## 1          mpg acceleration  0.42
## 2       weight acceleration -0.42
## 3    cylinders   model_year -0.35
## 4 displacement   model_year -0.37
```

```
## 5   horsepower   model_year -0.42
## 6       weight   model_year -0.31
## 7 acceleration   model_year  0.29
```

Take 0.3 as the threshold value, the following relationships may not be linear:

```
cor_table %>% dplyr::filter(abs(cor) < 0.3)
```

```
##           var1       var2  cor
## 1 acceleration model_year 0.29
```

v. Are there any pairs of independent variables that are highly correlated (r > 0.7)? ANSWER: The following relationships are highly correlated:

```
cor_table %>% dplyr::filter(abs(cor) >0.7)
```

```
##            var1         var2   cor
## 1           mpg     cylinders -0.78
## 2           mpg displacement -0.81
## 3     cylinders displacement  0.95
## 4           mpg   horsepower -0.78
## 5     cylinders   horsepower  0.84
## 6  displacement   horsepower  0.90
## 7           mpg       weight -0.83
## 8     cylinders       weight  0.90
## 9  displacement       weight  0.93
## 10   horsepower       weight  0.86
```

(b) linear regression model

```
regr_all <- lm(mpg~., data = auto_value)
summary(regr_all)
```

```
##
## Call:
## lm(formula = mpg ~ ., data = auto_value)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.6927 -2.3864 -0.0801  2.0291 14.3607
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.454e+01  4.764e+00  -3.051  0.00244 **
## cylinders    -3.299e-01  3.321e-01  -0.993  0.32122
## displacement  7.678e-03  7.358e-03   1.044  0.29733
## horsepower   -3.914e-04  1.384e-02  -0.028  0.97745
## weight       -6.795e-03  6.700e-04 -10.141  < 2e-16 ***
## acceleration  8.527e-02  1.020e-01   0.836  0.40383
## model_year    7.534e-01  5.262e-02  14.318  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.435 on 385 degrees of freedom
## Multiple R-squared:  0.8093, Adjusted R-squared:  0.8063
## F-statistic: 272.2 on 6 and 385 DF,  p-value: < 2.2e-16
```

i. Which independent variables have a 'significant' relationship with mpg at 1% significance?
ANSWER: The `weight`, `model_year` have a 'significant' relationship with mpg at 1% significance.

ii. Is it possible to determine which independent variables are the most effective at increasing mpg? If so, which ones, and if not, why not? (hint: units!) **ANSWER:** It seems `weight`, `model_year` are the most effective variables at increasing `mpg`.

(c)

```
auto_value_std <- data.frame(scale(auto_value))
auto_value_std$origin <-auto$origin
regr_std <- lm(mpg~., data = auto_value_std)
summary(regr_std)
```

i. Create fully standardized regression results: are these slopes easier to compare?

```
##
## Call:
## lm(formula = mpg ~ ., data = auto_value_std)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.15432 -0.26630 -0.01259  0.25440  1.71182
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.13213    0.03155  -4.187 3.50e-05 ***
## cylinders    -0.10703    0.07020  -1.524  0.12821
## displacement  0.32149    0.10261   3.133  0.00186 **
## horsepower   -0.08967    0.06761  -1.326  0.18549
## weight       -0.73028    0.07130 -10.243  < 2e-16 ***
## acceleration  0.02796    0.03472   0.805  0.42110
## model_year    0.36673    0.02444  15.005  < 2e-16 ***
## origin2       0.33696    0.07257   4.643 4.72e-06 ***
## origin3       0.36556    0.07082   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4236 on 383 degrees of freedom
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

ANSWER: The `origin` should not be standardize, and the slope are easier to compare with each other.

```
regr_cylinders <- lm(mpg ~ cylinders, data = auto_value_std)
regr_horsepower <- lm(mpg ~ horsepower, data = auto_value_std)
regr_acceleration <- lm(mpg ~ acceleration, data = auto_value_std)
summary(regr_cylinders)
```

ii. Regress mpg over each nonsignificant independent variable, individually. Which ones become significant when we regress mpg over them individually?

```
##
## Call:
```

```
## lm(formula = mpg ~ cylinders, data = auto_value_std)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.82463 -0.40784 -0.08113  0.32660  2.29555
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.731e-16  3.180e-02    0.00        1
## cylinders   -7.776e-01  3.184e-02  -24.43   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6295 on 390 degrees of freedom
## Multiple R-squared:  0.6047, Adjusted R-squared:  0.6037
## F-statistic: 596.6 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
summary(regr_horsepower)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = auto_value_std)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.73876 -0.41757 -0.04402  0.35401  2.16836
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.213e-16  3.175e-02    0.00        1
## horsepower  -7.784e-01  3.179e-02  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6285 on 390 degrees of freedom
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
summary(regr_acceleration)
```

```
##
## Call:
## lm(formula = mpg ~ acceleration, data = auto_value_std)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3048 -0.7195 -0.1536  0.6151  2.9775
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.427e-16  4.582e-02   0.000        1
## acceleration 4.233e-01  4.588e-02   9.228   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

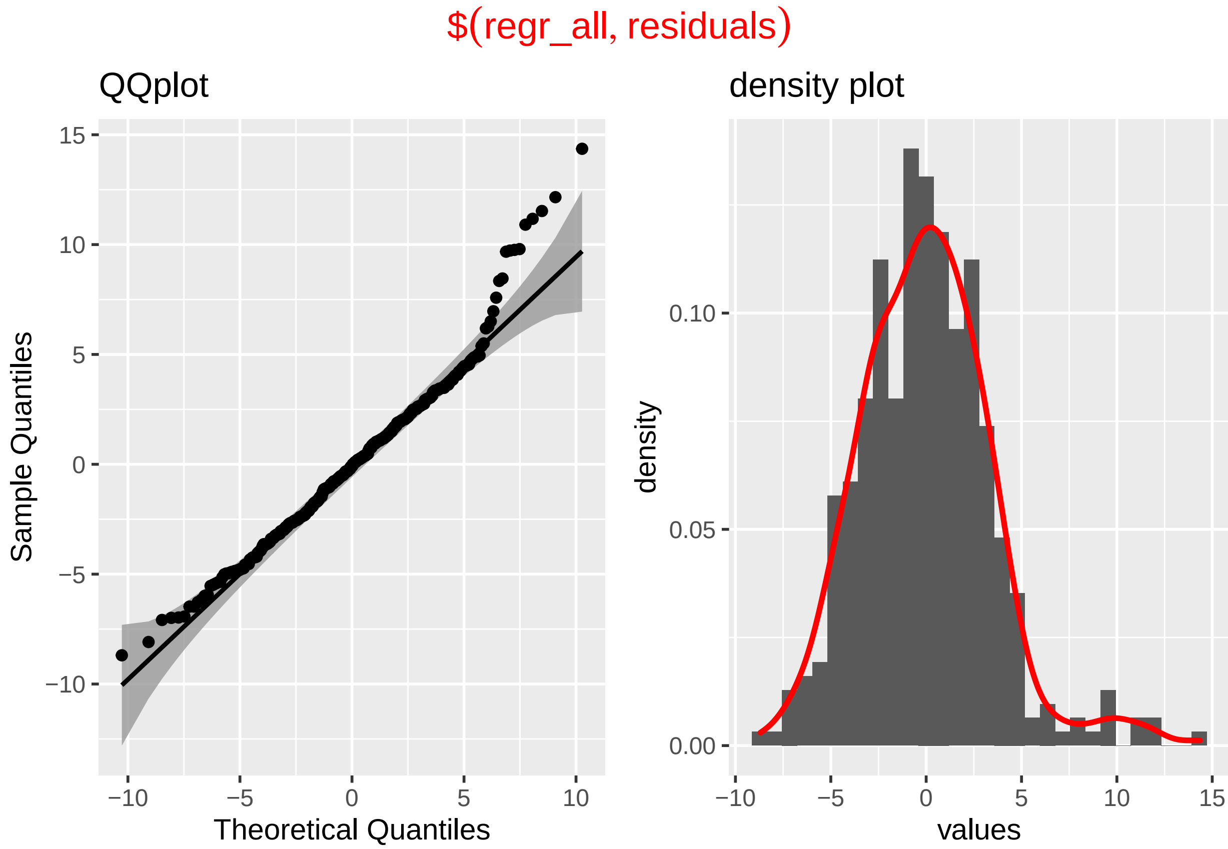
```
## Residual standard error: 0.9071 on 390 degrees of freedom
## Multiple R-squared:  0.1792, Adjusted R-squared:  0.1771
## F-statistic: 85.15 on 1 and 390 DF,  p-value: < 2.2e-16
```

ANSWER: When we regress `mpg` over each `cylinders`, `horsepower` and `acceleration`, individually, all nonsignificant independent variable become **significant**!

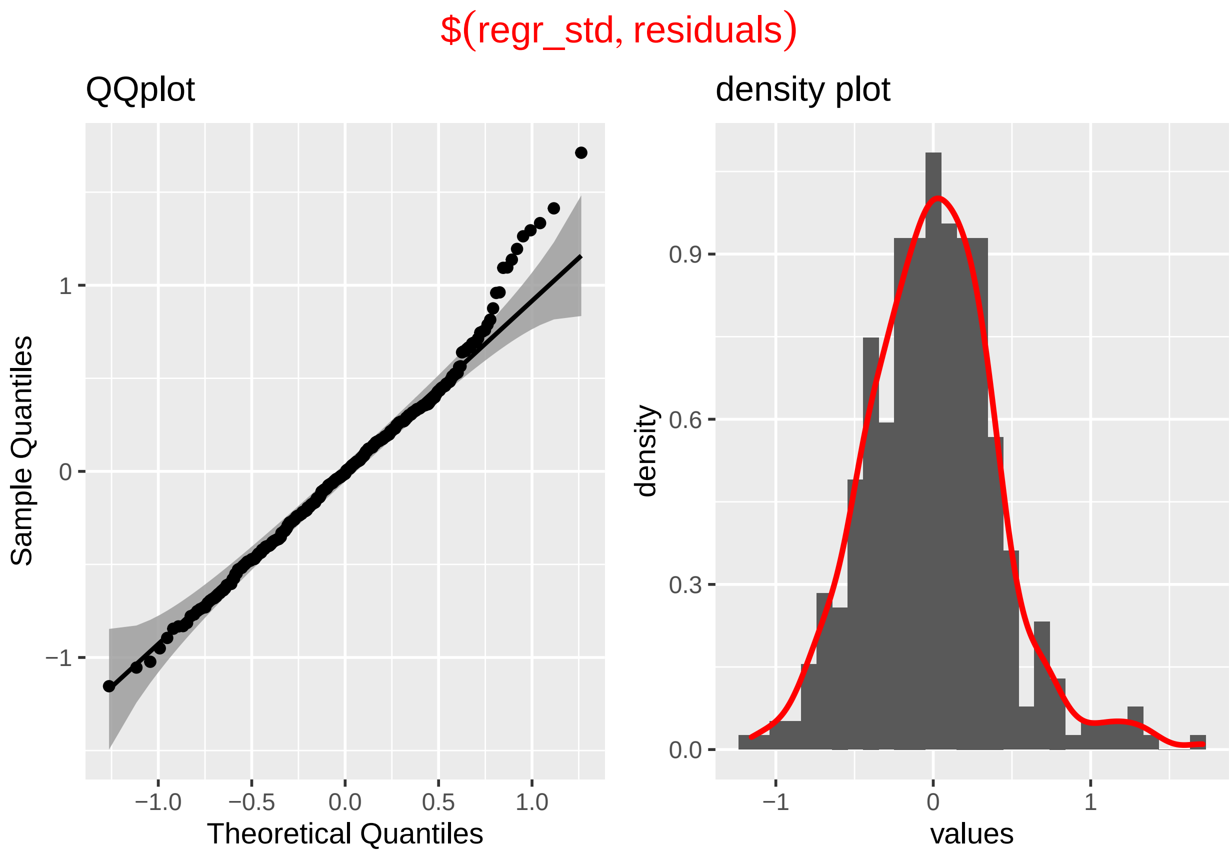
```r
# The function to plot qqplot
norm_qq_ggplot <- function(values){
  text <- substitute(values)
  df <- data.frame(value=values)
  gg <- ggplot(data = df, mapping = aes(sample = value)) +
      stat_qq_band() +
      stat_qq_line() +
      stat_qq_point() +
      labs(x = "Theoretical Quantiles", y = "Sample Quantiles", title = "QQplot")
  gg
}


density_hist_plot <- function(values){
  p <- ggplot(mapping = aes(values)) +
    geom_histogram(mapping = aes(y = stat(density))) +
    geom_density(color = "red", size = 1) +
    labs(title = "density plot")
  p
}

# combine two plots
density_qq_plot <- function(values){
  text <- substitute(values)
  p1 <- norm_qq_ggplot(values)
  p2 <- density_hist_plot(values)
  figure <- ggarrange(p1,p2)
  annotate_figure(figure,top = text_grob(text, color = "red", face = "bold", size = 14))
  # grid.arrange(p1,p2, nrow=1,ncol=2)
}
```

```r
density_qq_plot(regr_all$residuals)
```

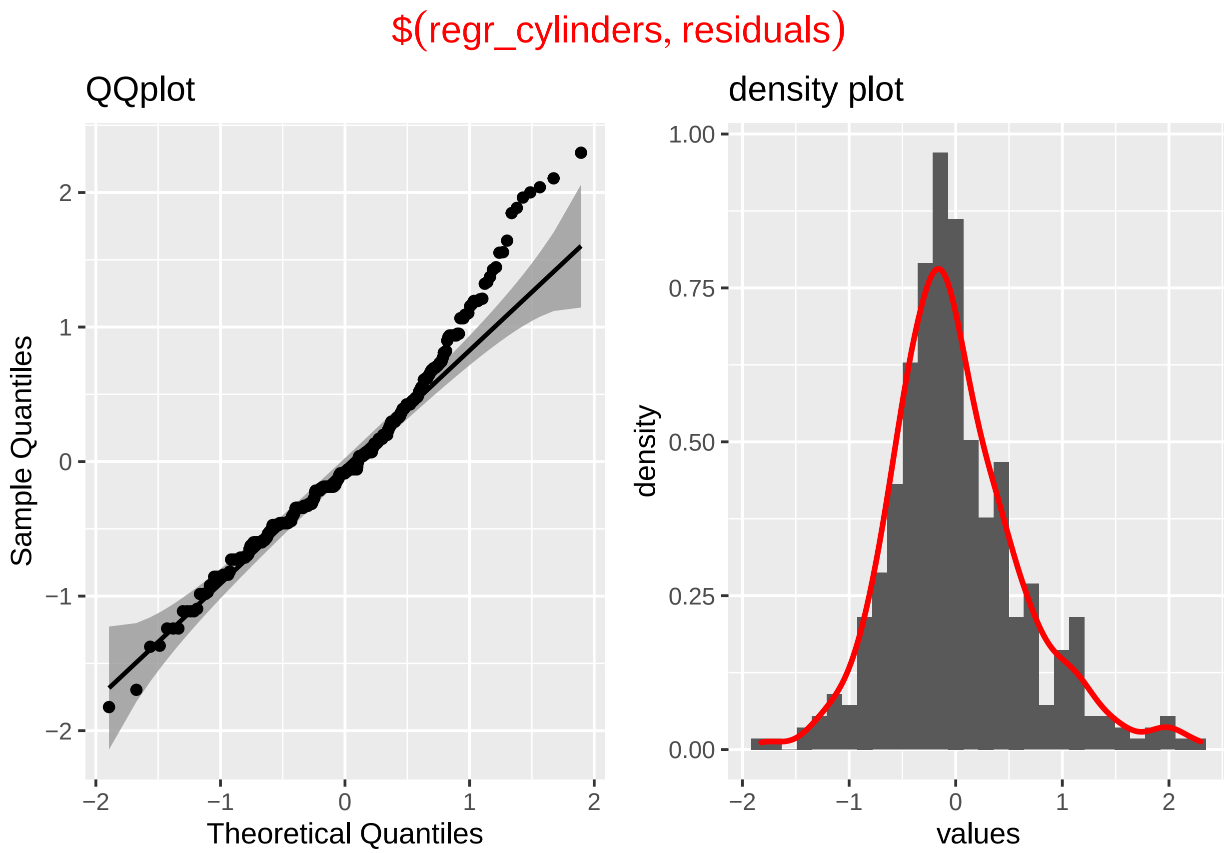
iii. Plot the density of the residuals: are they normally distributed and centered around zero?



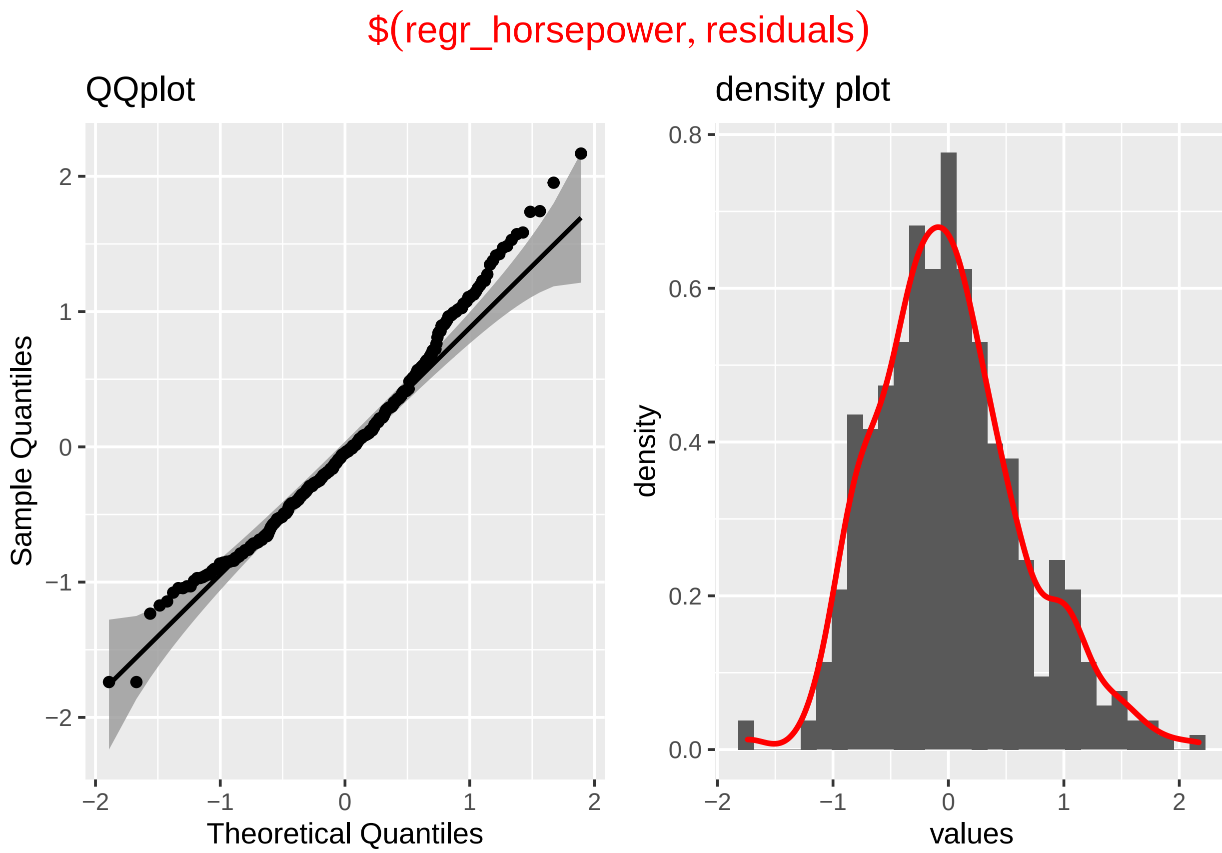

```
density_qq_plot(regr_std$residuals)
```



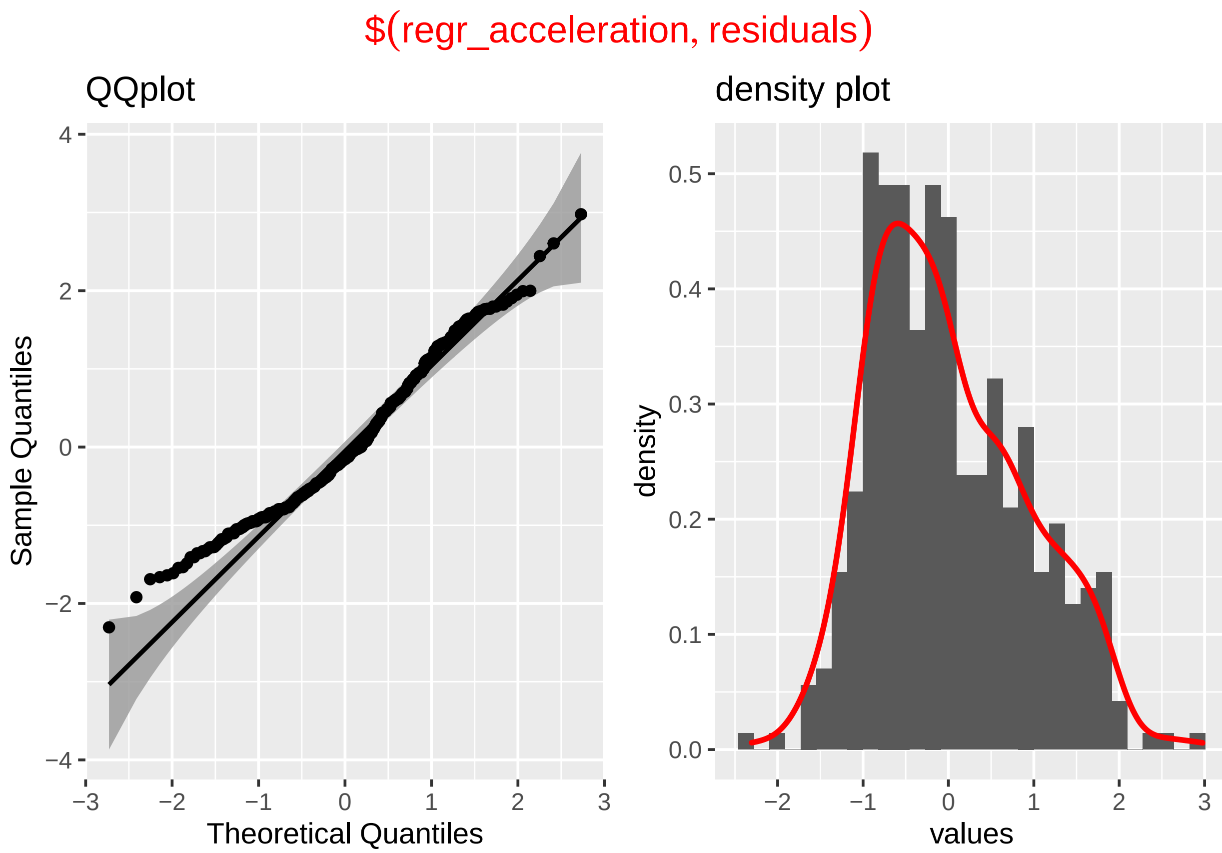

```
density_qq_plot(regr_cylinders$residuals)
```




```
density_qq_plot(regr_horsepower$residuals)
```




```
density_qq_plot(regr_acceleration$residuals)
```



ANSWER: All residuals are normally distributed and centered around zero.

Reference Link

- Counting the number of elements with the values of x in a vector
- R visulize
- Visualization of a correlation matrix using ggplot2
- Impute Missing Value
- How to convert integer to factor in R?
- ggplot2 violin plot
- Regularized Regression
- ggplot2 - Easy Way to Mix Multiple Graphs on The Same Page
- Extract variable names from list or vector in R