BACS HW10

109090046

2023-04-20

Helped by 109090035

Question 1)

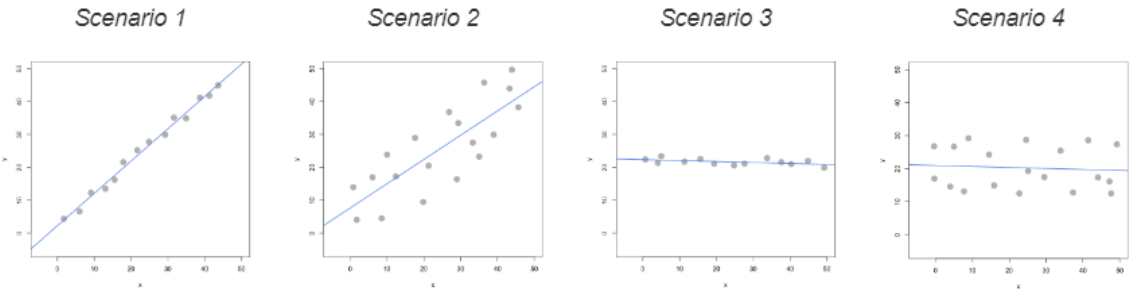
We will use the `interactive_regression()` function from `CompStatsLib` again – Windows users please make sure your desktop scaling is set to 100% and RStudio zoom is 100%; alternatively, run R from the Windows Command Prompt. To answer the questions below, understand each of these four scenarios by simulating them:

Scenario 1: Consider a very narrowly dispersed set of points that have a negative or positive steep slope

Scenario 2: Consider a widely dispersed set of points that have a negative or positive steep slope

Scenario 3: Consider a very narrowly dispersed set of points that have a negative or positive shallow slope

Scenario 4: Consider a widely dispersed set of points that have a negative or positive shallow slope



a.

Comparing scenarios 1 and 2, which do we expect to have a stronger R2?

ANS: We would expect Scenario 1 to have a stronger R2. In Scenario 1, the points are very narrowly dispersed around a steep slope, indicating a tight linear relationship between the variables. A higher R2 value represents a better fit of the regression line to the data points, and since the points are more closely clustered in Scenario 1, we can expect a higher R2 value.

b.

Comparing scenarios 3 and 4, which do we expect to have a stronger R2?

ANS: We would expect Scenario 3 to have a stronger R2. Even though the slope is shallow in both scenarios, the points are more narrowly dispersed around the slope in Scenario 3. This indicates a stronger linear relationship between the variables, leading to a higher R2 value.

c.

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

ANS: Intuitively, we would expect Scenario 1 to have a smaller SSE (sum of squared errors) because the points are more tightly clustered around the regression line, resulting in smaller errors. SSR (sum of squares due to regression) is likely to be larger in Scenario 1 since the points are more tightly clustered and better explained by the regression line. SST (total sum of squares) is the sum of SSE and SSR, so it depends on the specific data sets, but we could expect Scenario 1 to have a smaller SST due to the smaller dispersion of data points.

d.

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

ANS: Intuitively, we would expect Scenario 3 to have a smaller SSE because the points are more tightly clustered around the regression line, resulting in smaller errors. SSR is likely to be larger in Scenario 3 since the points are more tightly clustered and better explained by the regression line. As for SST, similar to the comparison between scenarios 1 and 2, it depends on the specific data sets, but we could expect Scenario 3 to have a smaller SST due to the smaller dispersion of data points.

Question 2)

Let's analyze the `programmer_salaries.txt` dataset we saw in class. Read the file using `read.csv("programmer_salaries.txt", sep="\t")` because the columns are separated by tabs (`\t`).

a.

Use the `lm()` function to estimate the regression model `Salary ~ Experience + Score + Degree` Show the beta coefficients, R2, and the first 5 values of y (`$fitted.values`) and (`$residuals`)

```
programmer_salaries <- read.csv("D:/下載/programmer_salaries.txt", sep="\t")

model <- lm(Salary ~ Experience + Score + Degree, data=programmer_salaries)

## Beta coefficients:

## (Intercept)  Experience       Score      Degree
##    7.944849    1.147582    0.196937    2.280424

##
## R-squared:

## [1] 0.8467961

##
## First 5 fitted values:

##        1        2        3        4        5
## 27.89626 37.95204 26.02901 32.11201 36.34251

##
## First 5 residuals:

##          1          2          3          4          5
## -3.8962605  5.0479568 -2.3290112  2.1879860 -0.5425072
```

b.

Use only linear algebra and the geometric view of regression to estimate the regression yourself:

i.

Create an X matrix that has a first column of 1s followed by columns of the independent variables(only show the code)

```
X <- programmer_salaries %$% cbind(1, Experience, Score, Degree)
```

ii.

Create a y vector with the Salary values (only show the code)

```
y <- programmer_salaries$Salary
```

iii.

Compute the beta_hat vector of estimated regression coefficients (show the code and values)

```
beta_hat <- solve(t(X) %*% X) %*% (t(X) %*% y)
beta_hat
```

```
##                [,1]
##            7.944849
## Experience 1.147582
## Score      0.196937
## Degree     2.280424
```

iv.

Compute a y_hat vector of estimated y values, and a res vector of residuals (show the code and the first 5 values of y_hat and res)

```
y_hat <- X %*% beta_hat
res <- y - y_hat
head(y_hat, 5)
```

```
##          [,1]
## [1,] 27.89626
## [2,] 37.95204
## [3,] 26.02901
## [4,] 32.11201
## [5,] 36.34251
```

```
head(res, 5)
```

```
##            [,1]
## [1,] -3.8962605
## [2,]  5.0479568
## [3,] -2.3290112
## [4,]  2.1879860
## [5,] -0.5425072
```

v.

Using only the results from (i) – (iv), compute SSR, SSE and SST (show the code and values)

```
SSR <- sum((y_hat - mean(y))^2)
SSE <- sum(res^2)
SST <- sum((y - mean(y))^2)
```

```
## SSR is: 507.896

## SSR is: 91.88949

## SSR is: 599.7855
```

c.

Compute R2 for in two ways, and confirm you get the same results (show code and values):

i.

Use any combination of SSR, SSE, and SST

```
R2_1 <- SSR / SST
R2_1

## [1] 0.8467961
```

ii.

Use the squared correlation of vectors y and y hat

```
R2_2 <- cor(y, y_hat)^2
R2_2

##           [,1]
## [1,] 0.8467961
```

Question 3)

We're going to take a look back at the early heady days of global car manufacturing, when American, Japanese, and European cars competed to rule the world. Take a look at the data set in file `auto-data.txt`. We are interested in explaining what kind of cars have higher fuel efficiency (mpg).

1. mpg: miles-per-gallon (dependent variable)
2. cylinders: cylinders in engine
3. displacement: size of engine
4. horsepower: power of engine
5. weight: weight of car
6. acceleration: acceleration ability of car
7. model_year: year model was released
8. origin: place car was designed (1: USA, 2: Europe, 3: Japan)
9. car_name: make and model names

Note that the data has missing values ('?' in data set), and lacks a header row with variable names:

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

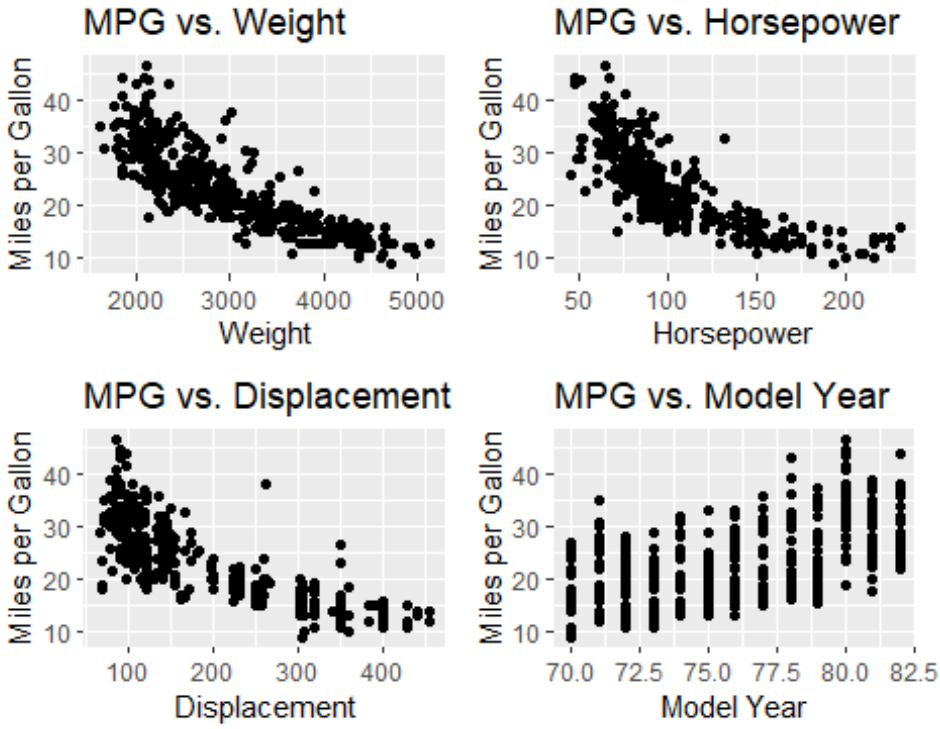
a.

Let's first try exploring this data and problem:

i.

Visualize the data as you wish (report only relevant/interesting plots)

```
library(patchwork)

## Warning: 套件 'patchwork' 是用 R 版本 4.2.3 來建造的

p1 <- ggplot(auto, aes(x = weight, y = mpg)) +
  geom_point() +
  theme_grey() +
  labs(title = "MPG vs. Weight", x = "Weight", y = "Miles per Gallon")

p2 <- ggplot(auto, aes(x = horsepower, y = mpg)) +
  geom_point() +
  theme_grey() +
  labs(title = "MPG vs. Horsepower", x = "Horsepower", y = "Miles per Gallon")

p3 <- ggplot(auto, aes(x = displacement, y = mpg)) +
  geom_point() +
  theme_grey() +
  labs(title = "MPG vs. Displacement", x = "Displacement", y = "Miles per Gallon")

p4 <- ggplot(auto, aes(x = model_year, y = mpg)) +
  geom_point() +
  theme_grey() +
  labs(title = "MPG vs. Model Year", x = "Model Year", y = "Miles per Gallon")

(p1 | p2) / (p3 | p4)

## Warning: Removed 6 rows containing missing values (`geom_point()`).
```



ii.

Report a correlation table of all variables, rounding to two decimal places (in the cor() function, set use="pairwise.complete.obs" to handle missing values)

```
cor_table <- cor(auto[,1:8], use = "pairwise.complete.obs")
round(cor_table, 2)
```

```
##                 mpg cylinders displacement horsepower weight acceleration
## mpg            1.00     -0.78        -0.80      -0.78  -0.83         0.42
## cylinders     -0.78      1.00         0.95       0.84   0.90        -0.51
## displacement  -0.80      0.95         1.00       0.90   0.93        -0.54
## horsepower    -0.78      0.84         0.90       1.00   0.86        -0.69
## weight        -0.83      0.90         0.93       0.86   1.00        -0.42
## acceleration   0.42     -0.51        -0.54      -0.69  -0.42         1.00
## model_year     0.58     -0.35        -0.37      -0.42  -0.31         0.29
## origin         0.56     -0.56        -0.61      -0.46  -0.58         0.21
##               model_year origin
## mpg                 0.58   0.56
## cylinders          -0.35  -0.56
## displacement       -0.37  -0.61
## horsepower         -0.42  -0.46
## weight             -0.31  -0.58
## acceleration        0.29   0.21
## model_year          1.00   0.18
## origin              0.18   1.00
```

iii.

From the visualizations and correlations, which variables appear to relate to mpg?

ANS: From the plots and correlation table, we can see that `weight`, `displacement`, `horsepower`, and `cylinders` have strong negative correlations with `mpg`. `Acceleration`, `model_year`, `origin` have weak positive correlations.

iv.

Which relationships might not be linear? (don't worry about linearity for rest of this HW)

ANS: `acceleration` and `model_year` might not be linear with `origin`.

v.

Are there any pairs of independent variables that are highly correlated (r > 0.7)?

ANS: Yes, such as: `displacement` and `cylinders`, `horsepower` and `cylinders`, `weight` and `cylinders`, `displacement` and `horsepower`, `displacement` and `weight`, `horsepower` and `weight`.

b.

Let's create a linear regression model where mpg is dependent upon all other suitable variables (Note: origin is categorical with three levels, so use `factor(origin)` in `lm(...)` to split it into two dummy variables)

```
regr <- lm(mpg ~ cylinders + displacement + horsepower + weight + acceleration + model_ye
ar + factor(origin), data = auto)

summary(regr)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + model_year + factor(origin), data = auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.0095 -2.0785 -0.0982  1.9856 13.3608
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.795e+01  4.677e+00  -3.839 0.000145 ***
## cylinders       -4.897e-01  3.212e-01  -1.524 0.128215
## displacement     2.398e-02  7.653e-03   3.133 0.001863 **
## horsepower      -1.818e-02  1.371e-02  -1.326 0.185488
## weight          -6.710e-03  6.551e-04 -10.243  < 2e-16 ***
## acceleration     7.910e-02  9.822e-02   0.805 0.421101
## model_year       7.770e-01  5.178e-02  15.005  < 2e-16 ***
## factor(origin)2  2.630e+00  5.664e-01   4.643 4.72e-06 ***
## factor(origin)3  2.853e+00  5.527e-01   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.307 on 383 degrees of freedom
##   (因為不存在，6 個觀察量被刪除了)
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

i.

Which independent variables have a 'significant' relationship with mpg at 1% significance?

ANS: `displacement`, `acceleration`, `model_year` have significant relationship with mpg at 1% significance, since p-values less than 0.01.

ii.

Looking at the coefficients, 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!)

ANS: It's difficult to compare the coefficients directly since they have different units. Standardizing the variables can help with comparison.

c.

Let's try to resolve some of the issues with our regression model above.

i.

Create fully standardized regression results: are these slopes easier to compare? (note: consider if you should standardize origin)

```
auto_standardized <- auto
auto_standardized[,1:7] <- scale(auto[,1:7])
regr_standardized <- lm(mpg ~ cylinders + displacement + horsepower + weight + accelerati
on + model_year + factor(origin), data = auto_standardized)
summary(regr_standardized)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders + displacement + horsepower + weight +
##     acceleration + model_year + factor(origin), data = auto_standardized)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.15270 -0.26593 -0.01257  0.25404  1.70942
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -0.13323    0.03174  -4.198 3.35e-05 ***
## cylinders       -0.10658    0.06991  -1.524  0.12821
## displacement     0.31989    0.10210   3.133  0.00186 **
## horsepower      -0.08955    0.06751  -1.326  0.18549
## weight          -0.72705    0.07098 -10.243  < 2e-16 ***
## acceleration     0.02791    0.03465   0.805  0.42110
## model_year       0.36760    0.02450  15.005  < 2e-16 ***
## factor(origin)2  0.33649    0.07247   4.643 4.72e-06 ***
## factor(origin)3  0.36505    0.07072   5.162 3.93e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.423 on 383 degrees of freedom
##   (因為不存在，6 個觀察量被刪除了)
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8205
## F-statistic: 224.5 on 8 and 383 DF,  p-value: < 2.2e-16
```

ANS: Yes, the slopes of the fully standardized regression results are easier to compare. When we standardize the variables, we put them on the same scale (mean = 0, standard deviation = 1), making it easier to interpret the effect of each variable on the dependent variable. This allows us to compare the relative importance of each independent variable in the regression model.

ii.

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

```
regr_cyl <- lm(mpg ~ cylinders, data = auto)
summary(regr_cyl)
```

```
##
## Call:
## lm(formula = mpg ~ cylinders, data = auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.2607  -3.3841  -0.6478   2.5538  17.9022
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  42.9493     0.8330   51.56   <2e-16 ***
## cylinders    -3.5629     0.1458  -24.43   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.942 on 396 degrees of freedom
## Multiple R-squared:  0.6012, Adjusted R-squared:  0.6002
## F-statistic: 597.1 on 1 and 396 DF,  p-value: < 2.2e-16
```

```
regr_hor <- lm(mpg ~ horsepower, data = auto)
summary(regr_hor)
```

```
##
## Call:
## lm(formula = mpg ~ horsepower, data = auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.5710  -3.2592  -0.3435   2.7630  16.9240
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 39.935861   0.717499   55.66   <2e-16 ***
## horsepower  -0.157845   0.006446  -24.49   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.906 on 390 degrees of freedom
##   (因為不存在，6 個觀察量被刪除了)
## Multiple R-squared:  0.6059, Adjusted R-squared:  0.6049
## F-statistic: 599.7 on 1 and 390 DF,  p-value: < 2.2e-16
```

```
regr_acc <- lm(mpg ~ acceleration, data = auto)
summary(regr_acc)
```

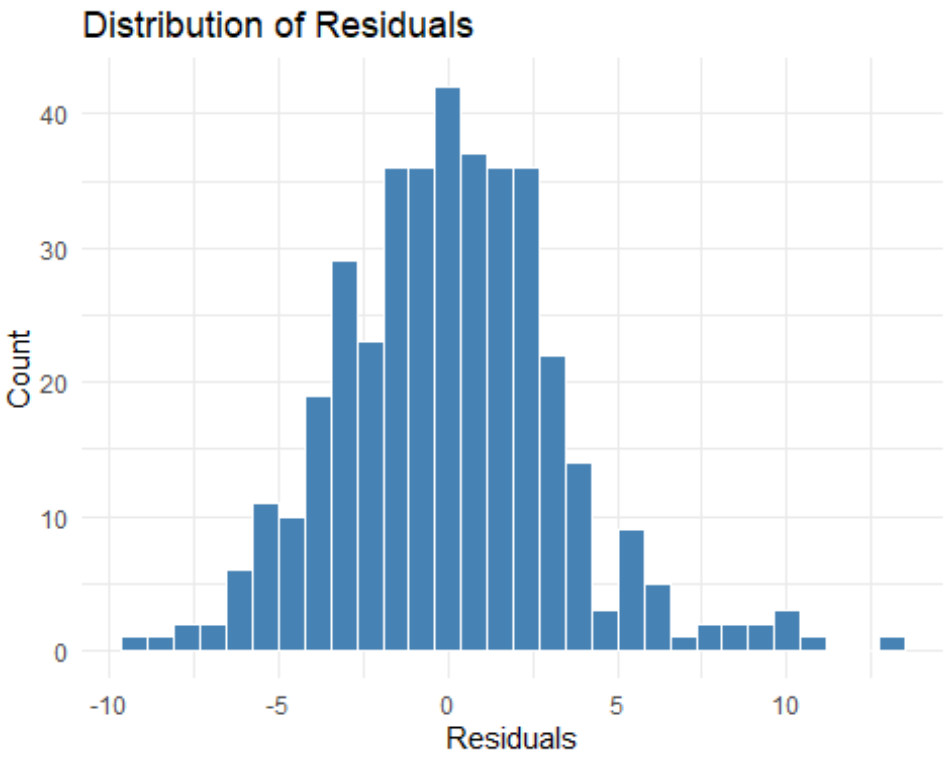
```
##
## Call:
## lm(formula = mpg ~ acceleration, data = auto)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -18.007  -5.636  -1.242   4.758  23.192
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.9698     2.0432   2.432   0.0154 *
## acceleration   1.1912     0.1292   9.217   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.101 on 396 degrees of freedom
## Multiple R-squared:  0.1766, Adjusted R-squared:  0.1746
## F-statistic: 84.96 on 1 and 396 DF,  p-value: < 2.2e-16
```

ANS: `acceleration` becomes significant because the p-value is 0.0154, larger than 0.01.

iii.

Plot the distribution of the residuals: are they normally distributed and centered around zero? (get the residuals of a fitted linear model, e.g. `regr <- lm(...)`, using `regr$residuals`

```
residuals_hist <- ggplot(data.frame(residuals = regr$residuals), aes(residuals)) +
  geom_histogram(fill = "steelblue", color = "white", bins = 30) +
  theme_minimal() +
  labs(title = "Distribution of Residuals", x = "Residuals", y = "Count")

residuals_hist
```



ANS: Yes, they are normally distributed and centered around zero.