Using R for data analysis SSA

Boerhaave Nascholing Leiden

May 17th, 2022

Important note

The primary goal of the self-study assignment (SSA) is to write an R Markdown document containing **R code** answering the questions below. Using the course and other on-line materials is permitted.

The steps you need to take:

1. Create a new R Markdown file.
2. Develop the code with your answers in the R Markdown file. Put each question in a separate section.
3. *When Knitting is possible:* Use `Knit` button regularly to check that your code generates the html report without any errors.

Diamonds dataset

You will analyse the `diamonds` dataset which contains various attributes of sold diamonds (see also `?diamonds`). The dataset comes with the `tidyverse` package. After you load the `tidyverse` library you will have access to the dataset in the `diamonds` variable. Make sure you put `library(tidyverse)` in the R chunk at the top of your R Markdown file.

```
library(tidyverse)
diamonds
```

```
# A tibble: 53,940 x 10
   carat cut       color clarity depth table price     x     y     z
   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
 7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
 8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
 9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
# ... with 53,930 more rows
```

Each row of the `diamonds` tibble describes one sold diamond. There are the following variables (columns):

- `price`: Price in US dollars.
- `carat`: Weight of the diamond (in carat units: 1 carat = 0.2g).
- `cut`: Quality of the cut (`Fair`, `Good`, `Very Good`, `Premium`, `Ideal`).
- `color`: Diamond colour, from `J` (worst) to `D` (best).
- `clarity`: How clear the diamond is (`I1` (worst), `SI2`, `SI1`, `VS2`, `VS1`, `VVS2`, `VVS1`, `IF` (best)).
- `x`, `y`, `z`: Length, width, depth. Each in mm.

- `depth`: Total depth percentage = z / mean(x, y) = 2 * z / (x + y).
- `table`: Width of top of diamond relative to widest point.

Questions

Q1. [0.5p] Show the type/class of the `diamonds` table. [0.5p] Show the type of the column `cut`.

```
class(diamonds)
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```
class(diamonds$cut)
```

```
[1] "ordered" "factor"
```

Q2. [1p] Show the structure of the `diamonds` table.

```
str(diamonds) # alternatively: glimpse(diamonds)
```

```
tibble [53,940 x 10] (S3: tbl_df/tbl/data.frame)
 $ carat  : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
 $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
 $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
 $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
 $ depth  : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
 $ table  : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
 $ price  : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
 $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
 $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
 $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

Q3. [1p] Print the last 7 rows of the `diamonds` table.

```
tail(diamonds,7)
```

```
# A tibble: 7 x 10
  carat cut       color clarity depth table price     x     y     z
  <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1  0.7  Very Good E     VS2      61.2    59  2757  5.69  5.72  3.49
2  0.72 Premium   D     SI1      62.7    59  2757  5.69  5.73  3.58
3  0.72 Ideal     D     SI1      60.8    57  2757  5.75  5.76  3.5
4  0.72 Good      D     SI1      63.1    55  2757  5.69  5.75  3.61
5  0.7  Very Good D     SI1      62.8    60  2757  5.66  5.68  3.56
6  0.86 Premium   H     SI2      61      58  2757  6.15  6.12  3.74
7  0.75 Ideal     D     SI2      62.2    55  2757  5.83  5.87  3.64
```

Q4. [1p] Calculate the median (number!) of the `depth` column from the `diamonds` table.

```
median(diamonds$depth)
```

```
[1] 61.8
```

Q5. [1p] Calculate and print the number of levels of the factor in the `cut` column.

```
nlevels(diamonds$cut) # alternatively: length(levels(diamonds$cut))
```

```
[1] 5
```

Q6. [3p] Make a `list` with two elements calculated as follows from the `diamonds` table. Name the first list element `maxPrice` and set it to the maximum diamond price. Name the second list element `colors` and

set it to the levels of the column `color`. Obviously, the first element should be a number and the second a character vector.

```
list(
  maxPrice = max( diamonds$price ),
  colors = levels( diamonds$color )
)
```

```
$maxPrice
[1] 18823

$colors
[1] "D" "E" "F" "G" "H" "I" "J"
```

Q7. Frequencies and cross table.

a) [1p] Count all the combinations of the value pairs in columns `cut` and `color`. This table should have three columns: `cut`, `color` and the number of occurrences.

```
diamonds %>% count(cut,color)
```

```
# A tibble: 35 x 3
   cut   color     n
   <ord> <ord> <int>
 1 Fair  D       163
 2 Fair  E       224
 3 Fair  F       312
 4 Fair  G       314
 5 Fair  H       303
 6 Fair  I       175
 7 Fair  J       119
 8 Good  D       662
 9 Good  E       933
10 Good  F       909
# ... with 25 more rows
```

b) [2p] Print a crosstable of `cut` and `color`, with `cut` categories given in columns.

```
diamonds %>%
  count(cut,color) %>%
  pivot_wider(names_from = cut, values_from = n)
```

```
# A tibble: 7 x 6
  color  Fair  Good `Very Good` Premium Ideal
  <ord> <int> <int>       <int>   <int> <int>
1 D       163   662        1513    1603  2834
2 E       224   933        2400    2337  3903
3 F       312   909        2164    2331  3826
4 G       314   871        2299    2924  4884
5 H       303   702        1824    2360  3115
6 I       175   522        1204    1428  2093
7 J       119   307         678     808   896
```

Q8. [3p] Group the `diamonds` table by `cut`. Summarise the mean `price` and the mean `carat` in each group.

```
diamonds %>%
  group_by(cut) %>%
  summarise(meanPrice=mean(price), meanCarat=mean(carat), .groups='drop')
```

```
# A tibble: 5 x 3
  cut       meanPrice meanCarat
  <ord>         <dbl>     <dbl>
1 Fair          4359.     1.05
2 Good          3929.     0.849
3 Very Good     3982.     0.806
4 Premium       4584.     0.892
5 Ideal         3458.     0.703
```

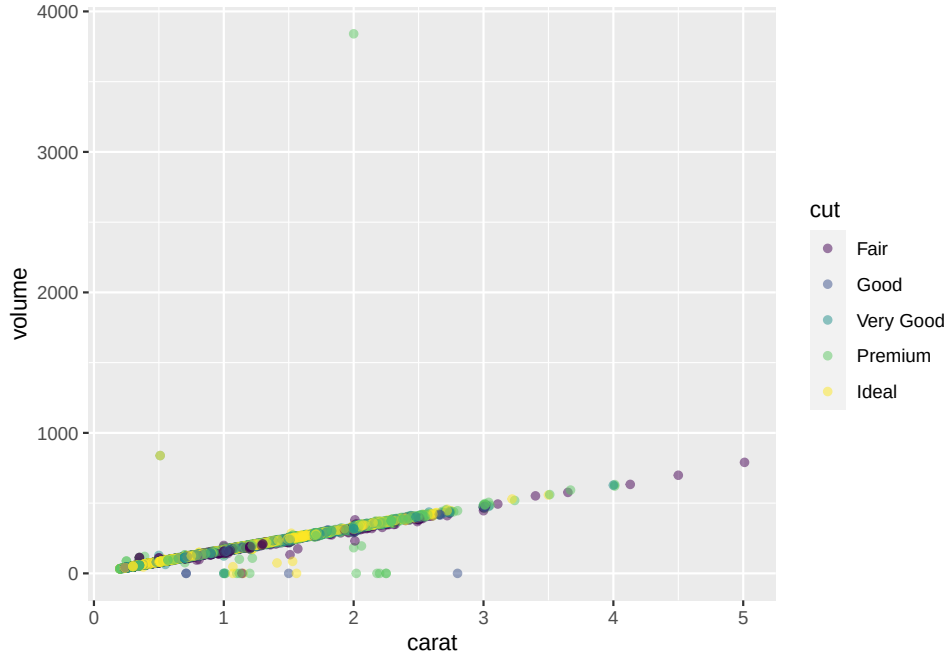
Q9. Diamond volume in a scatter plot.

a) [1p] Add a new column `volume` representing diamond's volume in cubic millimetres given the dimensions `x`, `y` and `z`. Store the tibble with the added column in a new variable `diamonds_volume`.

```
diamonds_volume <- diamonds %>% mutate(volume=x*y*z)
```

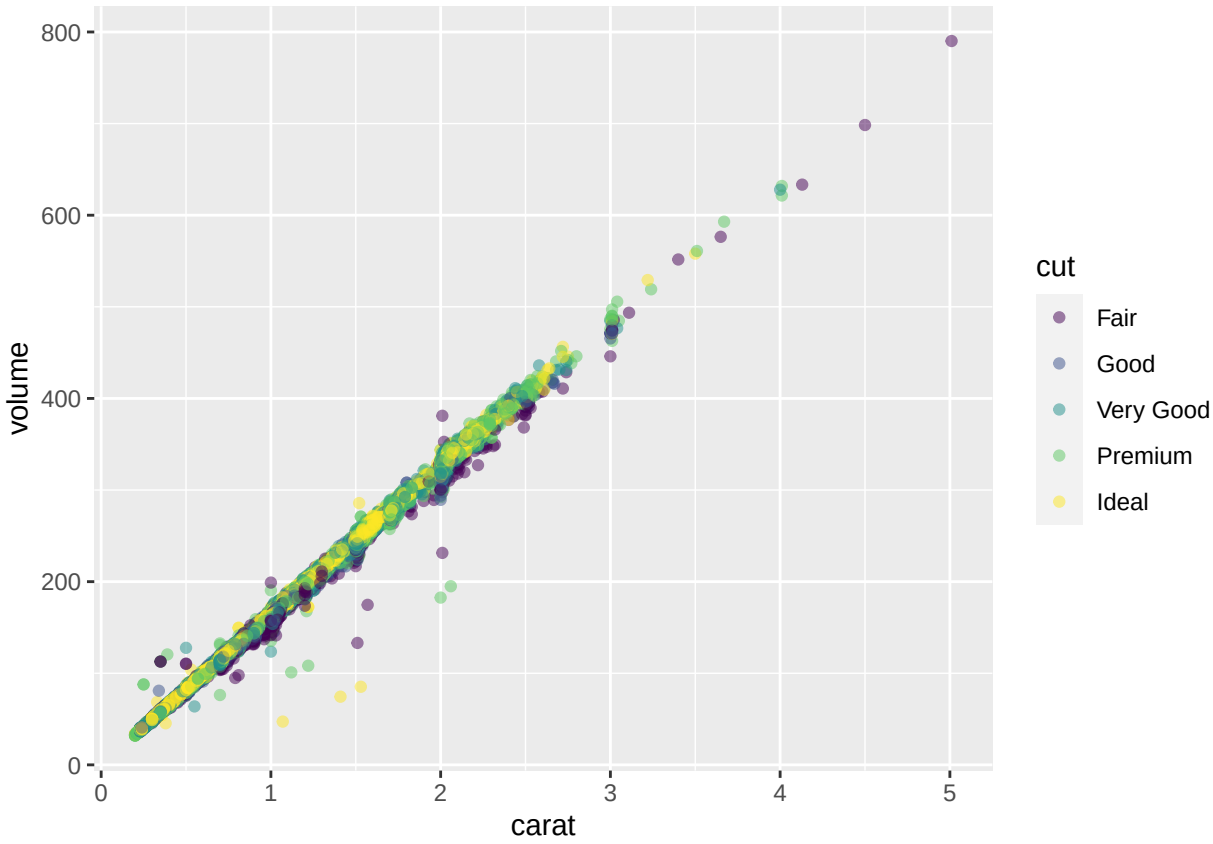
b) [2p] Use the data from `diamonds_volume` variable and plot the `volume` (vertical axis) against the `carat` (horizontal axis) in a scatterplot. Colour points by `cut`. Make points transparent (0.5).

```
ggplot(diamonds_volume) +
  aes(x = carat, y = volume, color = cut) +
  geom_point( alpha = 0.5 )
```



b) [1p] Replot the scatterplot in Q9.b but now with rows where $volume > 0$ and $volume \leq 800$.

```
ggplot(diamonds_volume %>% filter(volume > 0, volume <= 800)) +
  aes(x = carat, y = volume, color = cut) +
  geom_point( alpha = 0.5 )
```



Q10. Read/write CSV files.

a) [1p] Write the table `diamonds_volume` to a *comma-separated values* (CSV) file. Give the following name to the file: `diamonds_volume.csv`

```
write_csv(diamonds_volume, path = "diamonds_volume.csv")
```

```
Warning: The `path` argument of `write_csv()` is deprecated as of readr 1.4.0.
Please use the `file` argument instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

b) [1p] Read the file `diamonds_volume.csv` back into variable `d` and show it.

```
d <- read_csv(file = "diamonds_volume.csv")
d
```

```
# A tibble: 53,940 x 11
   carat cut       color clarity depth table price     x     y     z volume
   <dbl> <chr>     <chr> <chr>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43   38.2
 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31   34.5
 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31   38.1
 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63   46.7
 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75   51.9
 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48   38.7
 7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47   38.8
 8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53   42.3
 9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49   36.4
10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39   38.7
```

```
# ... with 53,930 more rows
```