Course of Study Bachelor Computer Science	Exercises Statistics WS 2020/21

Sheet I - Solutions

1 Descriptive Statistics - Variables

1. It is possible to transform a variable "downwards", from a scale with more information contained, to a scale with less information contained. Give an example for the variable *Price for a bottle of wine* for the transformation from a ratio to an ordinal scale.

   Answer:

   - Ratio scale: Price measured in euro.
   - Ordinal scale: Price mesured as cheap, normal, expensive.

2. Is it possible to transform a variable "upwards", from a scale with less information contained, to a scale with more information contained? Give an example (showing if it is possible or not)!

   Answer: No, you cannot gain more information about a variable by transforming the variable from one scale to another. Example: Assume that the variable *Price for a bottle of wine* originally has an ordinal scale. This means that you know for each bottle if it is cheap, normal or expensive. Given **only** this information for each bottle, do you know the price in euro (ratio scale) for the bottles? No!

3. Consider the question of describing students attitudes towards to legalisation of Marihuana, what proportion of them wants to legalize the drug and whether this proportion differs by gender and age.

   (a) Which data collection method is most suitable here: survey or experiment?

   (b) How could you capture the attitudes towards legalisation in a single variable?

   (c) Which variables are needed to answer the questions? Describe the type and the scale of the variables.

(d) How would an appropriate data set look? Try to describe the question in more details.

Answer:

(a) Survey: The information would be obtained via a questionaire given to a sample of students.

(b) There are different options to ask the students attitudes:

- simply ask: "What do you think about legalisation?"
  Problem: Capturing long answers in a variable attitutde may make it difficult to summarize and distil the information obtained.
- Common way: translate it into a score
  One could for example ask 5 "yes/no" questions which relate to attitudes towards legalisation like "Do you believe that legalisation would endanger the health of young people?", "Do you think legalization would encourage the entry into harder drugs?", ... The number of answers showing a positive attitude can be summed up. Thus the answers of each student can be summarized on a scale from 0 to 5.

(c) Needed variables are:

- Attitude: quantitative variable, ordinal scale
- Legalise: binary ("yes/no") variable capturing wether the student agrees to legalize Marihuana. This is qualititaive variable with nominal scale.
- Gender: qualititaive variable with nominal scale
- Age: quantitative (continous) variable with ratio scale.

(d) A data set might look as:

Student	A1	...	A5	Attitude	Legalize	Gender	Age
1	yes	...	no	3	yes	male	22
2	no	...	yes	2	no	female	25
⋮	⋮	...	⋮	⋮	⋮	⋮	⋮

A1, ..., A5 refer to variables capturing attitudes towards legalisiation and "Legalise" is the score variable summarizing these questions.
More detailled questions:

- What is the average attitude towards legalisation among students and how much does ist vary?

- What percentage of students answer “yes” when asked to legalise Marihuana?
- What is difference in the proportion calculated above when stratified by gender?
- What is average of those students who support the legalisation compared with the average age of those students who do not support the legalisation?

R Solutions Sheet 1: Introduction to R, RStudio

Dr. Falkenberg

WS 2020/21

Introduction to R, RStudio

Task 1: Calculate the following quantities:

```
# sum of 52.3, 74.8, 3.17
52.3+74.8+3.17
```

```
## [1] 130.27
```

```
# the square root of 144
144**0.5
```

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

```
# the 10-based logarithm of 200 multiplied with sin of $\pi/4$
log10(200)*sin(pi/4)
```

```
## [1] 1.627074
```

```
# the cumulative sum of the numbers 1,3,18,20,2
cumsum(c(1,3,18,20,2))
```

```
## [1]  1  4 22 42 44
```

```
# find 10 numbers between 0 and 20 rounded to the nearest
sample(x = 0:20, size = 10, replace = FALSE)
```

```
##  [1]  5  8 17 14  0 12  7  3  9  6
```

```
# or
round(runif(n = 10, min = 0, max = 20))
```

```
##  [1] 11 18  7 17 11 11 18 16 19 14
```

Task 2: Assigning Variables

```
# Assign the number 5 to x and the number 10 to y.
x <- 5
y <- 10
# Calculate the product of x and y.
x * y
```

```
## [1] 50
```

```
# Store the result in a new variable z.
z <- x * y
# Make a vector myvec of the objects x,y,z.
myvec <- c(x,y,z)
# Find the minimum, the maximum and the mean of the vector.
min(myvec)
```

```
## [1] 5
```

```
max(myvec)
```

```
## [1] 50
```

```
mean(myvec)
```

```
## [1] 21.66667
```

```
# Remove myvec from the workspace.
rm(myvec)
```

Task 3: The numbers below are the first ten days of rainfall in a year

```
# values: 0.1  0.5  2.3  1.1  11.3  14.7 23.4  15.7  0  0.9
# Read them into a vector using the c() command.
rainfall <- c(0.1,0.5,2.3,1.1,11.3,14.7,23.4,15.7,0,0.9)
# Calulate the mean and the standard deviation.
mean(rainfall)
```

```
## [1] 7
```

```
sd(rainfall)
```

```
## [1] 8.533594
```

```
# Calculate the cumulative rainfall over these ten days.
cumsum(rainfall)
```

```
##  [1]  0.1  0.6  2.9  4.0 15.3 30.0 53.4 69.1 69.1 70.0
```

```
# What is total sum of the rainfall?
sum(rainfall)
```

```
## [1] 70
```

```
# Which day saw the highest rainfall? Find an appropriate
# R command.
which.max(rainfall)
```

```
## [1] 7
```

```
# Take a subset of the rainfall data where rain is larger
# than 10.
rainfall[rainfall>10]
```

```
## [1] 11.3 14.7 23.4 15.7
```

```
# What is mean rainfall for days where the rainfall was at
# least 5?
mean(rainfall[rainfall >=5])
```

```
## [1] 16.275
```

```
# Subset the vector where it is either exactly 0 or 1.1 and
# find the corresponding days.
rainfall[rainfall == 0 | rainfall == 1.1]
```

```
## [1] 1.1 0.0
```

```
# alternative solution
rainfall[rainfall %in% c(0,1.1)]
```

```
## [1] 1.1 0.0
```

```
# days where the rainfall is 0 or 1.1
which(rainfall %in% c(0,1.1))
```

```
## [1] 4 9
```

Task 4: The length of five cylinders are 2.5, 3.4, 4.8, 3.1, 1.7 and their diameters are 0.7, 0.4, 0.5, 0.5, 0.9.

```
# Read these vectors into two vectors with appropriate names.
len <- c(2.5, 3.4, 4.8, 3.1, 1.7)
diam <- c(0.7, 0.4, 0.5, 0.5, 0.9)
# Calculate the volumes of each cylinder and store it
# in a new vector.
vol <- len * (diam/2)**2 * pi
vol
```

```
## [1] 0.9621128 0.4272566 0.9424778 0.6086836 1.0814933
```

```
# Assume the values are given in centimeter.
# Recalculate the volumes so that their units are cubic
# millimeter.
vol.cm <- 10*len * (10*diam/2)**2 * pi
vol.cm
```

```
## [1]  962.1128  427.2566  942.4778  608.6836 1081.4933
```

Task 5: Inspect the R commands union(), setdiff() and intersect() implying set operations.

```
# Make two vectors
x <- c(1,2,3,4,5)
y <- c(3,5,7,9)
# Find values that are contained in both x and y.
intersect(x,y)
```

```
## [1] 3 5
```

```
# Find values that are in x but not y and vice versa.
# x without y
setdiff(x,y)
```

```
## [1] 1 2 4
```

```
# y without x
setdiff(y,x)
```

```
## [1] 7 9
```

```
# Construct a vector that contains all values contained in
# either x or y. Compare the result with c(x,y).
# union(): values are either in x or y
union(x,y)
```

```
## [1] 1 2 3 4 5 7 9
```

```
# c(x,y) only concatenates x and y
c(x,y)
```

```
## [1] 1 2 3 4 5 3 5 7 9
```

Task 6: Construct a matrix with 8 rows and 10 columns. The first row should contain the numbers 0, 2, 4, …, 18 and the other rows should random integer numbers

between 0 and 100. Use runif() to create the random numbers and as.integer() to transform to integers.

```
mat1 <- matrix(c(seq(0,18, by = 2),
                 as.integer(runif(70,0,100))),
               nrow = 8, ncol = 10, byrow = TRUE)
mat1
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]    0    2    4    6    8   10   12   14   16    18
## [2,]   69   26    5   41    6   45    4   79    2    60
## [3,]   89   60   15   15   27   90   18   81   27    13
## [4,]   82   49   25    8   95   30   28    6   40    98
## [5,]   37   41    6   65   96   76    2    9   60    91
## [6,]   25   41   57    2   95   91   24   30   84    61
## [7,]   55   74    0   62   53   78   49   22    6    75
## [8,]   19   34   70   73   14   83   35   16   11    16
```

```
# Calculate the row means of this matrix and the standard
# deviation across the row means.
rm <- rowMeans(mat1)
rm
```

```
## [1]  9.0 33.7 43.5 46.1 48.3 51.0 47.4 37.1
```

```
sd(rm)
```

```
## [1] 13.63456
```

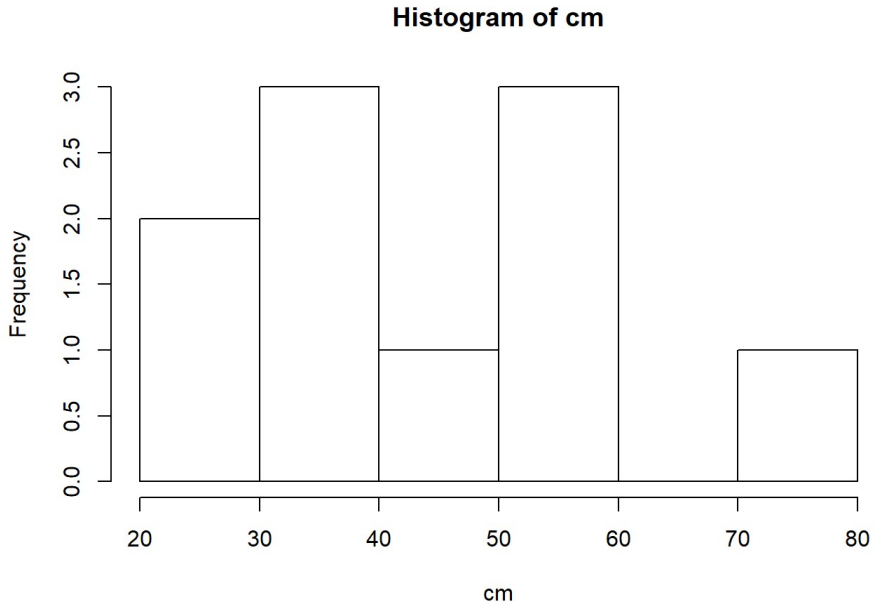
```
# Store the rows 2,3,..,8 in a other matrix and
# calculate the column means. Use the command hist()
# to create a histogram of the column means.
# removing the first row of mat1
mat2 <- mat1[-1,]
mat2
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]   69   26    5   41    6   45    4   79    2    60
## [2,]   89   60   15   15   27   90   18   81   27    13
## [3,]   82   49   25    8   95   30   28    6   40    98
## [4,]   37   41    6   65   96   76    2    9   60    91
## [5,]   25   41   57    2   95   91   24   30   84    61
## [6,]   55   74    0   62   53   78   49   22    6    75
## [7,]   19   34   70   73   14   83   35   16   11    16
```

```
# colum means of mat2
cm <- colMeans(mat2)
cm
```

```
##  [1] 53.71429 46.42857 25.42857 38.00000 55.14286 70.42857 22.85714 34.71429
##  [9] 32.85714 59.14286
```

```
# creating a histogram of cm
hist(cm)
```



Task 7: Inspect the R datasat mpg. Determine the types and the scales of measurement of all variables in the dataset mpg. Further more determine whether the variables are discret or continous.

```
# load packages
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------------------------- tidyverse 1.3.0
--
```

```
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## v purrr   0.3.3
```

```
## -- Conflicts ------------------------------------------------------------------------------ tidyverse_conflicts()
--
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
# display the variables and the first lines of the dataset
names(mpg)
```

```
##  [1] "manufacturer" "model"        "displ"        "year"         "cyl"
##  [6] "trans"        "drv"          "cty"          "hwy"          "fl"
## [11] "class"
```

```
head(mpg)
```

manufacturer <chr>	model <chr>	displ <dbl>	year <int>	cyl <int>	trans <chr>	drv <chr>	cty <int>	hwy <int>	fl <chr>
audi	a4	1.8	1999	4	auto(l5)	f	18	29	p
audi	a4	1.8	1999	4	manual(m5)	f	21	29	p
audi	a4	2.0	2008	4	manual(m6)	f	20	31	p
audi	a4	2.0	2008	4	auto(av)	f	21	30	p
audi	a4	2.8	1999	6	auto(l5)	f	16	26	p
audi	a4	2.8	1999	6	manual(m5)	f	18	26	p

6 rows | 1-10 of 11 columns

```
# description of the dataset
help(mpg)
```

```
## starting httpd help server ...
```

```
##  done
```

```
# create the tibble
str_mpg <- data_frame(name = names(mpg),
                      type = rep(NA,length(names(mpg))),
                      level = rep(NA,length(names(mpg))),
                      dc = rep(NA,length(names(mpg))))
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```
# properties of the variables
str_mpg[str_mpg$name =="manufacturer",2:4] <- c("qualitative","nominal","discrete")
str_mpg[str_mpg$name =="model",2:4] <- c("qualitative","nominal","discrete")
str_mpg[str_mpg$name =="displ",2:4] <- c("quantitative","ratio","continous")
str_mpg[str_mpg$name =="year",2:4] <- c("quantitative","interval","discrete")
str_mpg[str_mpg$name =="cyl",2:4] <- c("quantitative","ratio","discrete")
str_mpg[str_mpg$name =="trans",2:4] <- c("qualitative","nominal","discrete")
str_mpg[str_mpg$name =="drv",2:4] <- c("qualitative","nominal","discrete")
str_mpg[str_mpg$name =="cty",2:4] <- c("quantitative","ratio","continous")
str_mpg[str_mpg$name =="hwy",2:4] <- c("quantitative","ratio","continous")
str_mpg[str_mpg$name =="fl",2:4] <- c("qualitative","nominal","discrete")
str_mpg[str_mpg$name =="class",2:4] <- c("qualitative","nominal","discrete")
# display the structure
#str_mpg

# alternative solution

# create an empty tibble
str_mpg1 <- tibble(name = character(), type = character(), level = character(), dc = character())
# add rows
str_mpg1 <- str_mpg1 %>%
  add_row(name = "manufacturer", type = "qualitative", level = "nominal", dc = "discrete") %>%
  add_row(name = "model", type = "qualitative", level = "nominal", dc = "discrete") %>%
  add_row(name = "displ", type = "quantitative", level = "ratio", dc = "continous") %>%
  add_row(name = "year", type = "quantitative", level = "interval", dc = "discrete") %>%
  add_row(name = "cyl", type = "quantitative", level = "ratio", dc = "discrete") %>%
  add_row(name = "trans", type = "qualitative",level = "nominal", dc = "discrete") %>%
  add_row(name = "drv", type = "qualitative", level = "nominal", dc = "discrete") %>%
  add_row(name = "cty", type = "quantitative", level = "ratio", dc = "continous") %>%
  add_row(name = "hwy", type = "quantitative", level = "ratio", dc = "continous") %>%
  add_row(name = "fl", type = "qualitative", level = "nominal", dc = "discrete") %>%
  add_row(name = "class", type = "qualitative", level = "nominal", dc = "discrete")
#head(str_mpg1)

# all quant. and discrete variables
subset(str_mpg, subset = (type == "quantitative" & dc == "discrete"))
```

name <chr>	type <chr>	level <chr>	dc <chr>
year	quantitative	interval	discrete
cyl	quantitative	ratio	discrete

2 rows

Type, scale and discrete/continous for all columns of mpg

```
str_mpg
```

name <chr>	type <chr>	level <chr>	dc <chr>
manufacturer	qualitative	nominal	discrete
model	qualitative	nominal	discrete
displ	quantitative	ratio	continous
year	quantitative	interval	discrete

name <chr>	type <chr>	level <chr>	dc <chr>
cyl	quantitative	ratio	discrete
trans	qualitative	nominal	discrete
drv	qualitative	nominal	discrete
cty	quantitative	ratio	continous
hwy	quantitative	ratio	continous
fl	qualitative	nominal	discrete

1-10 of 11 rows Previous **1** 2 Next

Display all variables which are quantitative and discrete

```
subset(str_mpg, subset = (type == "quantitative" & dc == "discrete"))
```

name <chr>	type <chr>	level <chr>	dc <chr>
year	quantitative	interval	discrete
cyl	quantitative	ratio	discrete

2 rows