Programowanie w R - Lab 1

Dariusz Kopka gr.1 DS19

1 Zadanie 1

Proszę dla kredytu hipotecznego z ratą stałą udzielonego na kwotę K, przy rocznej stopie oprocentowania r zaciągniętego na okres n miesięcy wyznaczyć:

1. Wysokość oprocentowania miesięcznego: $q = 1 + \frac{r}{12}$
2. Wysokość raty miesięcznej: $R = K \cdot q^n \cdot \frac{q-1}{q^n-1}$
3. Całkowitą kwotę do spłaty: $F = R \cdot n$

```
# Wartości nie są przypadkowe
K <- 340124.79
r <- 7.46/100
n <- 274

q <- 1 + (r / 12)
R <- K * q**n * ((q - 1) / (q**n - 1))
F <- R * n
```

Ad. 1. Wysokość miesięcznego oprocentowania: $q = 1.006217$ %

Ad. 2. Rata miesięczna: $R = 2{,}588.16$ PLN

Ad. 3. Całkowita kwota do spłaty: $F = 709{,}155.42$ PLN

---

2 Zadanie 2

Proszę dla kredytu hipotecznego z ratą malejącą udzielonego na kwotę K, przy rocznej stopie oprocentowania r, zaciągniętego na okres n miesięcy wyznaczyć:

1. wysokość części kapitałowej raty $R_0 = \frac{K}{n}$,
2. wysokość części odsetkowej raty i-tej $R_1^i = \frac{(K-(i-1)\cdot R_0)\cdot r}{12}$,
3. wysokość raty i-tej $R^i = R_0 + R_1^i$,
4. całkowitą kwotę do spłaty $F = \sum_{i=1}^{n} R^i$.

Jaka była najniższa, średnia i najwyższa wartość raty?

```
R_0 <- K / n
iter <- seq(1, n)
R_1 <- ((K - (iter - 1) * R_0) * r) / 12
R_i <- R_0 + R_1
F_2 <- sum(R_i)
```

Ad. 1. Wysokość części kapitałowej raty: $R_0 = 1{,}241.33$ PLN

Ad. 2. i 3. Wysokość części odsetkowej raty R_1^i i raty całkowitej R_i:

i	Wysokość części odsedkowej	Wysokość raty
1	2,114.44 PLN	3,355.77 PLN
2	2,106.73 PLN	3,348.06 PLN
3	2,099.01 PLN	3,340.34 PLN
...	...	...
272	23.15 PLN	1,264.48 PLN
273	15.43 PLN	1,256.77 PLN
274	7.72 PLN	1,249.05 PLN

Ad. 4. Całkowita kwota do spłaty: $F = 630{,}860.63$ PLN

Wysokość najniższej/średniej/najwyższej raty: (1,249.05 / 2,302.41 / 3,355.77) PLN

3 Zadanie 3

W pliku `wig_changes.rds` znajduje się wektor uporządkowanych chronologicznie wartości tekstowych `+` oraz `-`, reprezentujących dni, w których:

- (+) wartość indeksu WIG wzrosła względem wartości z dnia poprzedniego,
- (−) wartość indeksu WIG zmalała względem wartości z dnia poprzedniego.

Bazując na tym wektorze wyznacz następującą macierz:

t-1\t	+	−
+	Prawdopodobieństwo wystąpienia stanu + po stanie +	Prawdopodobieństwo wystąpienia stanu − po stanie +
−	Prawdopodobieństwo wystąpienia stanu + po stanie −	Prawdopodobieństwo wystąpienia stanu − po stanie −

Podnieś utworzoną macierz do potęgi 3.

```
wig_changes <- readRDS('wig_changes.rds')
factors <- factor(wig_changes, levels = c("+", "-"))
from <- head(factors, -1)
to   <- tail(factors, -1)
# factors: c(- + - - + + - +)
# from:    c(- + - - + + -)
# to:      c(+ - - + + - +)
# Stwórz tabelę wszystkich możliwych przejść z `from` do `to`
trans_matrix <- table(from, to)
# proportions() przekształca tablicę liczebności w tablicę prawdopodobieństw
# margin = 1 - liczebności są zamieniane w prawdopodobieństwa względem _wierszy_.
# Tworzymy w ten sposób Macierz przejść Markowa
# (https://www.stat.auckland.ac.nz/~fewster/325/notes/ch8.pdf)
trans_prob <- proportions(trans_matrix, margin = 1)
trans_prob_txt <- capture.output(show(trans_prob))

trans_prob_3 <- trans_prob**3
trans_prob_3_txt <- capture.output(show(trans_prob_3))

result <- paste0("Macierz prawdopodobieństwa przejść:\n",
                   paste(trans_prob_txt, collapse = "\n"), "\n\n",
                 "Macierz prawdopodobieństwa do potęgi 3:\n",
                   paste(trans_prob_3_txt, collapse = "\n"))
cat(result)
```

```
## Macierz prawdopodobieństwa przejść:
##     to
## from         +         -
##    + 0.5443631 0.4556369
##    - 0.4998395 0.5001605
##
## Macierz prawdopodobieństwa do potęgi 3:
##     to
## from          +          -
##    + 0.16131177 0.09459249
##    - 0.12487965 0.12512042
```

4 Zadanie 4a

W ramach pewnego ubezpieczenia N klientów płaci składkę wysokości K w zamian za możliwość uzyskania kwoty F jeżeli nastąpi zdarzenie. Zbuduj symulację tego ubezpieczenia na okres T miesięcy zgodnie z poniższym algorytmem.

1. Przyjmij $t = 1$.
2. Wyznacz rezerwę na wypłaty: $S_t = \begin{cases} KN, & t = 1 \\ S_{t-1} + KN, & t > 1 \end{cases}$
3. Wyznacz liczbę wypłat: $a = \#\{n : c_n \geq F^{-1}_{t(2)}(0.9999)\}, \quad c_n \sim t(2)$.
4. Wypłać odszkodowania: $S_t = S_t - aF$
5. Sprawdź płynność: $S_t \geq 0$
    1. Jeżeli spełnione, to zmodyfikuj liczbę ubezpieczonych: $N = N + n - o - a$, gdzie n to losowa liczba z przedziału od 0 do 100 nowych klientów, a o to losowa liczba z przedziału od 0 do 90 klientów rezygnujących.
    2. Jeżeli nie spełnione, to firma zbankrutowała. Zatrzymaj algorytm przed czasem.
6. Przyjmij $t = t + 1$.
7. Jeżeli $t \leq T$, to przejdź do kroku 2, w przeciwnym przypadku: **KONIEC**.

```r
N <- 1000 # Liczba klientów
K <- 100 # Wysokość składki
F <- 200000 # Wysokość ubezpieczenia
T <- 100 # Symulacja na T miesięcy

t <- 1
s_t <- 0
log_df <- data.frame(
  t = integer(),
  N = integer(),
  S = numeric(),
  a = integer(),
  stringsAsFactors = FALSE
)

while (t <= T) {
    s_t <- s_t + K * N
    # Wygeneruj rozkład t-Studenta ze stopniem swobody = 2
    c_n <- rt(N, df = 2)
    # Kwantyl 0.9999 rozkładu t-Studenta
    th <- qt(0.9999, df = 2)
    a <- sum(c_n >= th)
    # Wypłata odszkodowań
    s_t <- s_t - a * F
    # Zapis stanu do tabeli
    log_df <- rbind(log_df, data.frame(t = t, N = N, S = s_t, a = a))

    if (s_t >= 0) {
```

```
        # Losowa liczba nowych klientów z przecdziału 0-100
        n <- sample(seq(0, 100), 1)
        # Losowa liczba rezygnujących klientów z przedziału 0-90
        o <- sample(seq(0, 90), 1)
        # Liczba likentów na następny okres (miesiąc)
        # Warto zauważyć, że odliczamy tych, którym firma
        # wypłaciła odszkodowanie.
        N <- N + n - o - a
    } else {
        # Firma zbankrutowała *KONIEC*
        break
    }
    t <- t + 1
}
```

---

5 Zadanie 4b

Wykonaj następujące czynności.

1. Stwórz funkcję przeprowadzającą zaprojektowaną symulację
    1. o argumentach K, N, F, T z wartościami domyślnymi,
    2. zwracającą wektor S_t długości T jako wynik. Jeżeli firma zbankrutowała, w wektorze powinny od tego momentu znajdować się wartości `NA`.
2. Napisz kod wykonujący symulację ubezpieczenia M razy i zapisujący wyniki do macierzy `SIM` postaci:

		Miesiąc symulacji				
		1	2	3	...	T
Numer symulacji	1					
	2	Rezerwy w miesiącu 1 symulacji 2				
	3					
	...					
	M					

3. Bazując na macierzy `SIM`, odpowiedz na poniższe pytania.
    1. Jakie jest prawdopodobieństwo tego, że spółka nie zbankrutuje do chwili $t = 1, 2, \ldots, T$?
    2. Jaki średni poziom rezerw będzie miała spółka pod warunkiem, że nie zbankrutuje do chwili $t = 1, 2, \ldots, T$?
    3. Jaki jest oczekiwany okres życia spółki przy założeniu, że maksymalny czas jej życia wynosi T?

```
N <- 1600 # Liczba klientów
K <- 100 # Wysokość składki
F <- 500000 # Wysokość ubezpieczenia
T <- 100 # Symulacja na T miesięcy
M <- 50 # Liczba symulacji

t <- 1
s_t <- 0

fn.simulate <- function(M) {
    t <- 1
    s_t <- 0
    row <- rep(NA, T)
    while (t <= T) {
        s_t <- s_t + K * N
        c_n <- rt(N, df = 2)
        th <- qt(0.9999, df = 2)
        a <- sum(c_n >= th)
        s_t <- s_t - a * F

        if (s_t >= 0) {
            n <- sample(seq(0, 100), 1)
            o <- sample(seq(0, 90), 1)
            N <- N + n - o - a
```

```
        } else {
            # Bankrut!
            # cat("Firma zbanrutowała w miesiącu (", t, "), przy (", a,") zdarzeniach.\n")
            break
        }
        row[t] <- s_t
        t <- t + 1
    }
    return(row)
}

symulacja <- lapply(seq(1:M), FUN = fn.simulate)
symulacja_mat <- do.call(rbind, symulacja)
rownames(symulacja_mat) <- paste0("Sym_", seq(1:M) )
colnames(symulacja_mat) <- paste0("Miesiac_", seq(1:T) )
knitr::kable(symulacja_mat[, 1:7],  booktabs = FALSE, row.names = TRUE)
```

	Miesiac_1	Miesiac_2	Miesiac_3	Miesiac_4	Miesiac_5	Miesiac_6	Miesiac_7
Sym_1	160000	323500	488200	648000	806500	467600	133100
Sym_2	160000	319400	478000	640600	804900	966900	1129300
Sym_3	NA	NA	NA	NA	NA	NA	NA
Sym_4	160000	317400	473100	626700	283400	448400	612800
Sym_5	160000	316700	469800	622900	774700	935200	1100900
Sym_6	160000	321700	483100	638800	801100	970500	1135800
Sym_7	160000	315000	473600	139300	306700	478600	651000
Sym_8	160000	NA	NA	NA	NA	NA	NA
Sym_9	NA	NA	NA	NA	NA	NA	NA
Sym_10	160000	327000	NA	NA	NA	NA	NA
Sym_11	160000	NA	NA	NA	NA	NA	NA
Sym_12	160000	317600	474700	635000	796800	961100	1123100
Sym_13	160000	326300	493600	666700	835400	1005500	1172300
Sym_14	160000	324900	491500	666900	844500	1024400	1207100
Sym_15	160000	NA	NA	NA	NA	NA	NA
Sym_16	160000	319300	469800	617500	773200	926100	1076500
Sym_17	160000	320700	484200	645100	808200	972300	1140900
Sym_18	160000	326100	495600	163800	332700	497900	669500
Sym_19	160000	320700	484600	648800	810800	975600	1147400
Sym_20	160000	320000	478800	635300	793000	457300	626100
Sym_21	160000	316300	479300	642100	809500	475000	636100
Sym_22	160000	321400	487500	654300	816600	982300	1148600
Sym_23	160000	NA	NA	NA	NA	NA	NA
Sym_24	160000	319500	476100	633900	787200	934000	1084800
Sym_25	160000	324200	481200	137500	NA	NA	NA
Sym_26	160000	323100	492500	668000	835400	1005400	1174400
Sym_27	160000	316000	474200	632700	788300	942300	1093900
Sym_28	160000	320000	476600	629200	781700	932400	1079300
Sym_29	160000	321600	NA	NA	NA	NA	NA
Sym_30	160000	324200	485700	652700	318000	478700	631900
Sym_31	160000	326900	492500	658500	327300	497400	670000
Sym_32	160000	316400	471100	632300	793500	960100	628300
Sym_33	160000	317500	476900	637800	794600	951200	1105700
Sym_34	160000	319700	482600	644500	808300	968500	1129700

	Miesiac_1	Miesiac_2	Miesiac_3	Miesiac_4	Miesiac_5	Miesiac_6	Miesiac_7
Sym_35	160000	319600	477400	637000	791700	946600	1096600
Sym_36	160000	NA	NA	NA	NA	NA	NA
Sym_37	160000	317900	478700	643800	803100	955600	608100
Sym_38	160000	320900	NA	NA	NA	NA	NA
Sym_39	160000	320500	484100	643900	310000	473100	639200
Sym_40	160000	NA	NA	NA	NA	NA	NA
Sym_41	160000	NA	NA	NA	NA	NA	NA
Sym_42	160000	319700	NA	NA	NA	NA	NA
Sym_43	160000	317100	473800	623300	774300	923000	1068900
Sym_44	160000	NA	NA	NA	NA	NA	NA
Sym_45	160000	321100	NA	NA	NA	NA	NA
Sym_46	160000	321400	479700	634900	792500	945400	1099900
Sym_47	160000	322700	491700	662300	826200	492700	663500
Sym_48	160000	326700	498400	668600	836700	502800	668400
Sym_49	160000	312900	469400	625000	774000	423100	565400
Sym_50	160000	319900	475000	624700	780900	436000	595400

```r
prawd_istnienia <- 100 - (100 * colSums(is.na(symulacja_mat)) / M)

srednie_rezerwy <- colMeans(symulacja_mat, na.rm = TRUE)

czas_zycia <- apply(symulacja_mat, 1, function(x) {
  first_na <- which(is.na(x))[1]
  if (is.na(first_na)) {
    return(ncol(symulacja_mat))  # przeżyła do końca
  } else {
    return(first_na - 1)  # bankructwo w miesiącu `first_na`
  }
})

oczekiwany_czas_zycia <- mean(czas_zycia)
```

Miesiąc życia	Prawdopodobieństwo istnienia (w %)	Średnie rezerwy
1	96	160000
2	80	320588
3	70	481229
4	70	599497
5	68	715344
6	68	773912
7	68	891585
8	66	1015345
9	66	1085582
10	66	1141924
11	66	1243055
12	66	1344861
13	66	1477158
14	66	1534173
15	62	1733968
…	…	…
86	60	8238387

Miesiąc życia	Prawdopodobieństwo istnienia (w %)	Średnie rezerwy
87	60	8426033
88	60	8547647
89	60	8619407
90	60	8657970
91	60	8729773
92	60	8869003
93	60	8991043
94	60	9063390
95	60	9136077
96	60	9260080
97	60	9384797
98	60	9509767
99	60	9635437
100	60	9761957

Oczekiwany okres życia spółki wynosi 61.62 miesięcy.

---

6 Zadanie 5

Plik `age.rds` zawiera dane dotyczące wieku klientów pewnego banku. Przeanalizuj te dane pod kątem odpowiedzi na następujące pytania.

1. Jaki wiek ma najmłodszy i najstarszy klient?
2. Jaki jest przeciętny wiek klientów banku?
3. Jak bardzo zróżnicowani są klienci banku pod względem wieku?
4. Ilu klientów banku jest niepełnoletnich? Jaki to procent całości?
5. Ilu klientów banku jest w wieku 30–50 lat? Jaki to procent całości?
6. Ilu klientów nie podało swojego wieku? Jaki to procent całości?
7. Ile klientów bank posiada w segmentach wiekowych $[16, 17]$, $[18, 24]$, $[25, 34]$, $[35, 44]$, $[45, 64]$, $[65, \infty)$? Jaki to procent całości?

```r
df <- readRDS("age.rds")
age <- na.omit(df)
najmlodszy <- min(age)
najstarszy <- max(age)
przecietny <- mean(age)
odchylenie <- sd(age)
rozstep <- IQR(age)

niepelnoletni <- sum(age < 18)
niepelnoletni_p <- 100 * niepelnoletni / length(age)
niepelnoletni_pa <- 100 * niepelnoletni / length(df)
klienci_30_50 <- sum(age >= 30 & age < 51)
klienci_30_50_p <- 100 * klienci_30_50 / length(age)
klienci_30_50_pa <- 100 * klienci_30_50 / length(df)
klienci_na <- length(df) - length(age)
klienci_na_p <- 100 * klienci_na / length(df)

przedzialy <- c(16, 18, 25, 35, 45, 65, Inf)
# Dla liczb całkowitych przedział [16, 17] jest tożsamy z [16, 18)
#                       podobnie [18, 24] jest tożsamy z [18, 25)
tablica <- table(cut(age, breaks = przedzialy, include.lowest = TRUE, right = FALSE))
tablicaA <- 100 * proportions(tablica)
tablicaB <- 100 * tablica / length(df)
```

Ad. 1. Najmłodszy klient ma 16 lat, najstarszy 86.

Ad. 2. Przeciętny wiek klienta banku wynosi 44.5549412 (średnia wieku).

Ad. 3. Odchylenie standardowe wynosi 10.0046376 a rozstęp międzykwartylowy (IQR) wynosi 13.

Ad. 4. Klientów niepełnoletnich jest 33, co stanowi 0.33 % wszystkich klientów oraz 0.3317583 % klientów, którzy podali wiek.

Ad. 5. Klientów w wieku 30-50 lat jest 6536, co stanowi 65.36 % wszystkich klientów, oraz 65.7082537 % klientów, którzy podali wiek.

Ad. 6. Wieku nie podało 53 klientów, co stanowi 0.53 % wszystkich klientów.

Ad. 7. W segmentach wiekowych $[16, 17]$, $[18, 24]$, $[25, 34]$, $[35, 44]$, $[45, 64]$, $[65, \infty)$, które dla wieku są tożsame z segmentami prawostronnie otwartymi $[16, 18)$ $[18, 25)$ $[25, 35)$ $[35, 45)$ $[45, 65)$ $[65, \infty]$ liczba klientów wynosi odpowiednio:

```
##
##  [16,18)  [18,25)  [25,35)  [35,45)  [45,65) [65,Inf]
##       33      192     1387     3336     4765      234
```

Procentowo względem klientów, którzy podali wiek:

```
##
##  [16,18)  [18,25)  [25,35)  [35,45)  [45,65) [65,Inf]
##     0.33     1.93    13.94    33.54    47.90     2.35
```

Procentowo względem wszystkich klientów banku:

```
##
##  [16,18)  [18,25)  [25,35)  [35,45)  [45,65) [65,Inf]
##     0.33     1.92    13.87    33.36    47.65     2.34
```

---

7 Zadanie 6

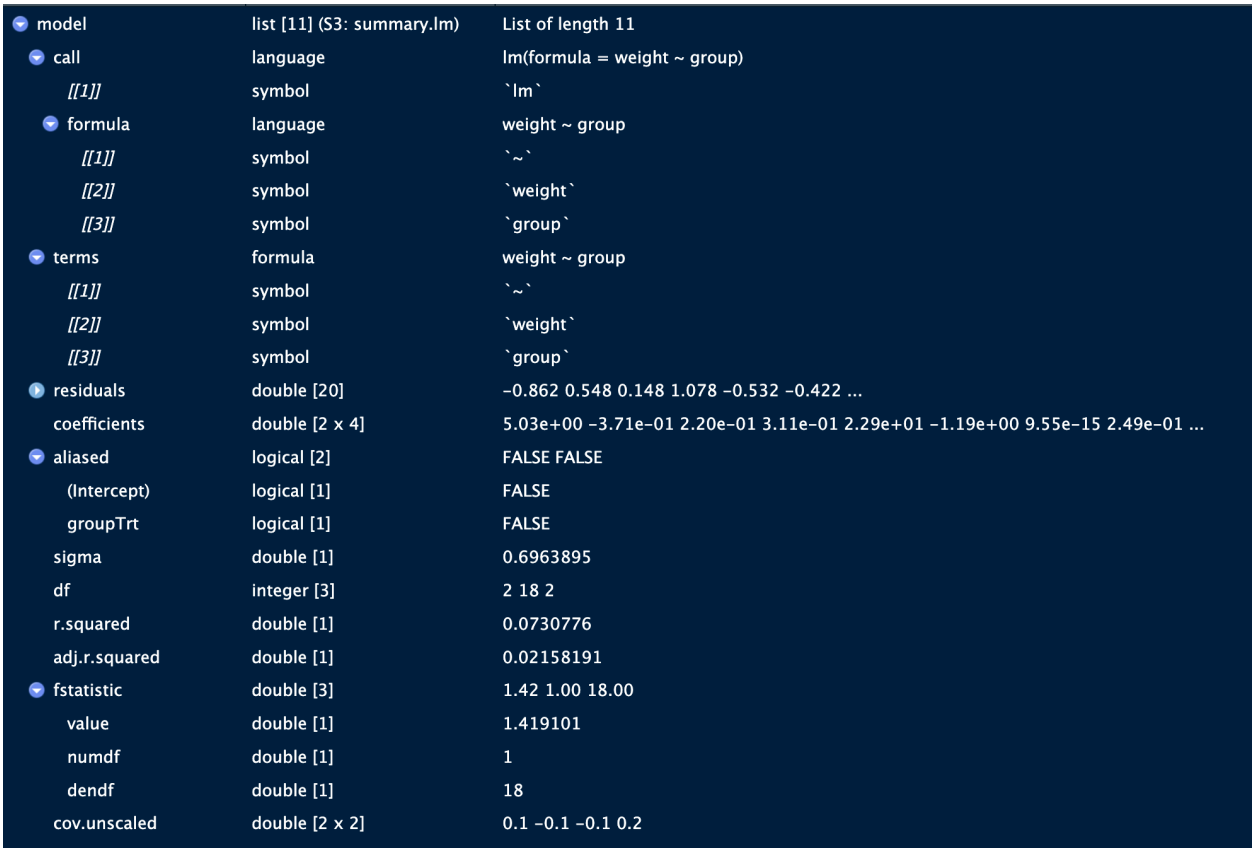
Wykonanie poniższego kodu spowoduje skonstruowanie prostego modelu liniowego zapisanego w postaci listy w obiekcie model. Wykonaj ten kod, a następnie:

1. przyjrzyj się strukturze obiektu model,
2. znajdź i wyświetl współczynniki modelu (`coefficients`),
3. znajdź i wyświetl wartości resztowe modelu (`residuals`),
4. znajdź i wyświetl wartość dopasowanego R^2 (`adj.r.squared`).

```
c(4.17,5.58,5.18,6.11,4.50,4.61,5.17,4.53,5.33,5.14) -> ctl
c(4.81,4.17,4.41,3.59,5.87,3.83,6.03,4.89,4.32,4.69) -> trt
gl(2, 10, 20, labels = c("Ctl","Trt")) -> group
c(ctl, trt) -> weight
summary(lm(weight ~ group)) -> model
```

7.0.1 Ad. 1. przyjrzyj się strukturze obiektu model

```
knitr::include_graphics("images/model_struktura.png")
```



7.0.2 Ad. 2. znajdź i wyświetl współczynniki modelu (`coefficients`)

```
model$coefficients
```

```
##             Estimate Std. Error  t value     Pr(>|t|)
## (Intercept)    5.032  0.2202177 22.85012 9.547128e-15
## groupTrt      -0.371  0.3114349 -1.19126 2.490232e-01
```

7.0.3 Ad. 3. znajdź i wyświetl wartości resztowe modelu (`residuals`)

```
model$residuals
```

```
##      1      2      3      4      5      6      7      8      9     10     11
## -0.862  0.548  0.148  1.078 -0.532 -0.422  0.138 -0.502  0.298  0.108  0.149
##     12     13     14     15     16     17     18     19     20
## -0.491 -0.251 -1.071  1.209 -0.831  1.369  0.229 -0.341  0.029
```

7.0.4 Ad. 4. znajdź i wyświetl wartość dopasowanego R^2 (`adj.r.squared`)

```
model$adj.r.squared
```

```
## [1] 0.02158191
```

---

8 Zadanie 7

Załaduj plik `ugly_diamonds.csv` do R w postaci poprawnej ramki danych, tzn. ramki danych która spełnia poniższą specyfikację:

```
'data.frame':10 obs. of 10 variables:
$ carat : num 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23
$ cut : chr "Ideal" "Premium" "Good" "Premium" ...
$ color : chr "E" "E" "E" "I" ...
$ clarity: chr "SI2" "SI1" "VS1" "VS2" ...
$ depth : num 61.5 59.8 56.9 62.4 NA 62.8 62.3 61.9 65.1 59.4
$ table : int 55 61 65 58 58 57 57 55 61 61
$ price : int 326 326 327 334 335 336 336 337 337 338
$ x : num 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4
$ y : num 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05
$ z : num 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39
```

```
diamonds <- read.csv("ugly_diamonds.csv", header = TRUE, sep = "%", skip = 4)

fn.convert_comma <- function(M) {
  M <- gsub(",", ".", M, fixed = TRUE)
  as.numeric(M)
}

cols_to_fix <- c("carat", "depth", "price", "x", "y", "z")
diamonds[cols_to_fix] <- lapply(diamonds[cols_to_fix], fn.convert_comma)
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion
```

```
diamonds$price <- as.integer(diamonds$price)

str(diamonds)
```

```
## 'data.frame':    10 obs. of  10 variables:
##  $ carat  : num  0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23
##  $ cut    : chr  "Ideal" "Premium" "Good" "Premium" ...
##  $ color  : chr  "E" "E" "E" "I" ...
##  $ clarity: chr  "SI2" "SI1" "VS1" "VS2" ...
##  $ depth  : num  61.5 59.8 56.9 62.4 NA 62.8 62.3 61.9 65.1 59.4
##  $ table  : int  55 61 65 58 58 57 57 55 61 61
##  $ price  : int  326 326 327 334 335 336 336 337 337 338
##  $ x      : num  3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4
##  $ y      : num  3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05
##  $ z      : num  2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39
```

---

9 Zadanie 8

Plik `bank_register.rds` zawiera dane dotyczące klientów pewnego banku w postaci następującej struktury.

id	date	income	demographic	products
Identyfikator klienta i umowy.	*Data dokonania wpisu.*	*Roczny dochód klienta.*	*Płeć, wiek i liczba dzieci.*	*Posiadane obecnie produkty bankowe.*
463_1	Jul 21, 2018	15.331,22$	F,37,0	DEP,MOR

Przekształć te dane do poniższej postaci.

client_id	agreement_id	date	income	sex	age	child	dep	cre	mor
Id klienta.	*Id umowy.*	*Data.*	*Dochód.*	*Płeć.*	*Wiek.*	*Dzieci.*	*Czy posiada depozyt?*	*Czy posiada kredyt?*	*Czy posiada kredyt hipoteczny?*
463	1	2018-06-21	15331.22	F	37	0	TRUE	FALSE	TRUE

```
bank_register <- readRDS("bank_register.rds")

frame <- data.frame(
    date = as.Date(bank_register$date, format = "%b %d, %Y")
)
frame[c("client_id", "agreement_id")] <- do.call(rbind,
                                                 strsplit(bank_register$id, '_' ))
frame[c("sex", "age", "child")] <- do.call(rbind,
                                           strsplit(bank_register$demographic, ',' ))

frame$income <- as.numeric(gsub(",", ".",
                                gsub("\\.", "",
                                     gsub("\\$", "", bank_register$income))))

frame$mor <- grepl("MOR", bank_register$products, ignore.case = TRUE)
frame$dep <- grepl("DEP", bank_register$products, ignore.case = TRUE)
frame$cre <- grepl("CRE", bank_register$products, ignore.case = TRUE)

columns <- c("client_id", "agreement_id", "date", "income",
             "sex", "age", "child", "dep", "cre", "mor")
output <- frame[1:10, columns]

# Tabela wejściowa - dziesięć pierwszych rzędów
knitr::kable(bank_register[1:10, ], booktabs = TRUE, row.names = TRUE)
```

	id	date	income	demographic	products
1	463_1	Jul 21, 2018	15.331,22$	F,37,0	
2	12_2	Dec 20, 2018	29.326,10$	M,33,1	
3	1_3	Dec 13, 2018	65.221,96$	M,74,3	DEP
4	368_4	Oct 15, 2018	92.897,54$	F,29,3	CRE
5	421_5	Jan 24, 2018	49.513,16$	F,64,0	
6	14_6	Jun 19, 2018	95.417,87$	M,16,0	DEP,MOR
7	301_7	Nov 01, 2018	53.220,92$	M,40,0	
8	491_8	May 30, 2018	15.651,98$	M,36,0	MOR,CRE

	id	date	income	demographic	products
9	448_9	Oct 06, 2018	83.196,39$	F,20,2	MOR
10	884_10	Sep 11, 2018	90.831,43$	M,19,0	MOR,DEP

```
# Tabela wyjściowa - dziesięć pierwszych rzędów
knitr::kable(output[1:10, ], booktabs = TRUE, row.names = )
```

client_id	agreement_id	date	income	sex	age	child	dep	cre	mor
463	1	2018-07-21	15331.22	F	37	0	FALSE	FALSE	FALSE
12	2	2018-12-20	29326.10	M	33	1	FALSE	FALSE	FALSE
1	3	2018-12-13	65221.96	M	74	3	TRUE	FALSE	FALSE
368	4	2018-10-15	92897.54	F	29	3	FALSE	TRUE	FALSE
421	5	2018-01-24	49513.16	F	64	0	FALSE	FALSE	FALSE
14	6	2018-06-19	95417.87	M	16	0	TRUE	FALSE	TRUE
301	7	2018-11-01	53220.92	M	40	0	FALSE	FALSE	FALSE
491	8	2018-05-30	15651.98	M	36	0	FALSE	TRUE	TRUE
448	9	2018-10-06	83196.39	F	20	2	FALSE	FALSE	TRUE
884	10	2018-09-11	90831.43	M	19	0	TRUE	FALSE	TRUE

10 Zadanie 9

Plik `albums.csv` zawiera następujące dane dotyczące albumów muzycznych:

- `artist_id` – identyfikator artysty,
- `album_title` – tytuł albumu,
- `genre` – gatunek muzyczny,
- `year_of_pub` – rok publikacji,
- `num_of_tracks` – liczba piosenek na płycie,
- `num_of_sales` – liczba sprzedanych płyt,
- `rolling_stone_critic` – ocena Rolling Stone Magazine,
- `mtv_critic` – ocena MTV,
- `music_maniac_critic` – ocena Music Maniac.

Bazując na zdobytej dotychczas wiedzy przeprowadź prostą analizę tej próbki.

```
suppressPackageStartupMessages(library(dplyr))
suppressPackageStartupMessages(library(ggplot2))

# Wczytywanie danych
albums <- read.csv("albums.csv")

# Najlepiej sprzedające się albumy
top_sold_albums <- albums %>%
    group_by(artist_id, album_title) %>%
    reframe(artist_id, album_title, genre, year_of_pub, num_of_sales) %>%
    arrange(desc(num_of_sales))
knitr::kable(top_sold_albums[1:10, ], booktabs = TRUE,
             row.names = TRUE, caption = "Najlepiej sprzedające się albumy")
```

Table 6: Najlepiej sprzedające się albumy

	artist_id	album_title	genre	year_of_pub	num_of_sales
1	12047	Decent Waterbuck	Heavy Metal	2010	999994
2	4842	Little Of Pahari	Hard Rock	2009	999991
3	25396	Gabby Malay American Boy Sudden	K-Pop	2019	999977
4	37484	Every Country Toys Chameleon	Rock	2010	999976
5	8123	cuddly Trick Sudden Javanese Kia Hills	Unplugged	2012	999969
6	9202	Algerian Color Sufficient	Rap	2012	999958
7	5330	Move Like a Seat	Heavy Metal	2004	999950
8	23827	Competitive Severe Inside My Girls Lion	Heavy Metal	2016	999925
9	730	Cow Matteo High Computer Systems	Indie	2009	999924
10	4073	Try Rich Marketing	Jazz	2009	999915

```
# Top 10 gatunków muzycznych wg sprzedaży
top_genres_by_sales <- albums %>%
    group_by(genre) %>%
    summarise(total_sales = sum(num_of_sales)) %>%
```

```
    arrange(desc(total_sales))
knitr::kable(top_genres_by_sales[1:10, ], booktabs = TRUE,
             row.names = TRUE, caption = "Top 10 gatunków wg sprzedaży")
```

Table 7: Top 10 gatunków wg sprzedaży

	genre	total_sales
1	Indie	4699805459
2	Pop	3916637999
3	Rap	2888914186
4	Latino	1953132011
5	Pop-Rock	1939002474
6	Punk	1914131397
7	Rock	1899856670
8	Dance	1882029421
9	Compilation	1009992493
10	Gospel	1005454870

```
# Top 10 gatunków muzycznych wg ilości albumów
top_genres_by_number <- albums %>%
    group_by(genre) %>%
    summarise(total_albums = n()) %>%
    arrange(desc(total_albums))
knitr::kable(top_genres_by_number[1:10, ], booktabs = TRUE,
             row.names = TRUE, caption = "Top 10 gatunków wg ilości albumów")
```

Table 8: Top 10 gatunków wg ilości albumów

	genre	total_albums
1	Indie	9384
2	Pop	7755
3	Rap	5788
4	Latino	3898
5	Pop-Rock	3880
6	Rock	3804
7	Punk	3787
8	Dance	3775
9	Gospel	2008
10	Compilation	2003

```
# Top 10 tytułów albumów
top_album_names <- albums %>%
    group_by(album_title) %>%
    summarize(album_count = n()) %>%
    arrange(desc(album_count))
knitr::kable(top_album_names[1:10, ], booktabs = TRUE,
             row.names = TRUE, caption = "Top 10 tytułów albumów muzycznych")
```

Table 9: Top 10 tytułów albumów muzycznych

	album_title	album_count
1	Of Love	211
2	In the Sky	124
3	Loves	111
4	Next to	88
5	Of	87
6	Noodles Moves In the Sky Ultra-Maximus	85
7	The	84
8	Vampire Chip A Quiet	83
9	China	82
10	Afrodite Potion	70

```
# Najbardziej płodni artyści
top_artists <- albums %>%
    group_by(artist_id) %>%
    summarize(sum_tracks = sum(num_of_tracks), sum_albums = n(), sum_sales = sum(num_of_sales))

# - po ilości sprzedanych płyt
top_artists %>%
    arrange(desc(sum_sales)) %>%
    reframe(artist_id = artist_id, sum_sales = sum_sales) %>%
    head(10) %>%
    knitr::kable(booktabs = TRUE, row.names = TRUE,
                 caption = "Najlepiej sprzedający się artyści")
```

Table 10: Najlepiej sprzedający się artyści

	artist_id	sum_sales
1	11290	6267844
2	33896	6072907
3	44614	5643206
4	7889	5573891
5	49450	5401584
6	5508	5329169
7	27648	5267296
8	40017	5214678
9	22799	5196851
10	15528	5193117

```
# - po ilości wydanych albumów
top_artists %>%
    arrange(desc(sum_albums)) %>%
    reframe(artist_id = artist_id, sum_albums = sum_albums) %>%
    head(10) %>%
    knitr::kable(booktabs = TRUE, row.names = TRUE,
                 caption = "Artyści, którzy wydali najwięcej płyt")
```

Table 11: Artyści, którzy wydali najwięcej płyt

	artist_id	sum_albums
1	10708	11
2	1608	9
3	4044	9
4	8330	9
5	11106	9
6	11290	9
7	22058	9
8	29669	9
9	33896	9
10	36534	9

```
# Szukamy korelacji między polami w tabeli
korelacja <- cor(albums[, c("num_of_sales", "rolling_stone_critic", "mtv_critic", "music_maniac_critic"]
knitr::kable(korelacja)
```

	num_of_sales	rolling_stone_critic	mtv_critic	music_maniac_critic
num_of_sales	1.0000000	-0.0024935	0.0005231	-0.0011949
rolling_stone_critic	-0.0024935	1.0000000	0.0011983	-0.0012693
mtv_critic	0.0005231	0.0011983	1.0000000	-0.0018090
music_maniac_critic	-0.0011949	-0.0012693	-0.0018090	1.0000000

Próbka zawiera 100000 albumów 43225 artystów wydanych od 2000 do 2019 roku w 38 gatunkach.

Macież korelacji nie wykazała liniowych zależności ani pomiędzy ocenami a ilością sprzedanych płyt, ani między samymi ocenami.

Dane wyglądają na wygenerowane losowo.
