Analiza danych ankietowych - sprawozdanie 1

"Lista zadań nr 1"

Przedmiot i prowadzący:
Analiza danych ankietowych,
poniedziałki 9.15 - 11.00 (grupa nr 2),
Inż. Hubert Woszczek

Aleksandra Hodera (268733) Aleksandra Polak (268786)

Spis treści

1 Wstęp

W poniższym sprawozdaniu zostały przedstawione wyniki listy zadań nr 1 przygotowanej w ramach laboratoriów z analizy danych ankietowych prowadzonych przez inż. Huberta Woszczek do wykładu dr inż. Aleksandry Grzesiek.

2 Użyte biblioteki

W tym punkcie zostały przedstawione wszystkie biblioteki, które użyłyśmy podczas tworzenia raportu:

```
library(ggplot2)
library(tidyr)
library(dplyr)
library(kableExtra)
library(vcd)
library(knitr)
library(likert)
library(reshape2)
library(stats)
library(binom)
library(GenBinomApps)
```

3 Część I

3.1 Zadanie 1

W pewnej dużej agencji reklamowej przeprowadzono ankietę mającą na celu ocenę poziomu satysfakcji z pracy. Wzięło w niej udział dwieście losowo wybranych osób (losowanie proste ze zwracaniem). W pliku "ankieta.csv"umieszczono odpowiedzi na kilka z zadanych pytań:

- "W jakim dziale jesteś zatrudniony? zmienna **DZIAŁ** przyjmująca wartości: **HR** (Dział obsługi kadrowo-płacowej), **IT** (Dział utrzymania sieci i systemów informatycznych), **DK** (Dział Kreatywny) lub **DS** (Dział Strategii).
- "Jak długo pracujesz w firmie? zmienna **STAŻ** przyjmująca wartości: **1** (poniżej jednego roku), **2** (między jednym rokiem a trzema latami) lub **3** (powyżej trzech lat).
- "Czy pracujesz na stanowisku menedżerskim? zmienna **CZY_KIER** przyjmująca wartości: **Tak** (stanowisko menedżerskie) lub **Nie** (stanowisko inne niż menedżerskie).
- "Jak bardzo zgadzasz się ze stwierdzeniem, że firma pozwala na elastyczne godziny pracy tym samym umożliwiając zachowanie równowagi między pracą, a życiem prywatnym? zmienna **PYT_1** przyjmująca wartości: **-2** (zdecydowanie się nie zgadzam), **-1** (nie zgadzam się), **0** (nie mam zdania), **1** (zgadzam się), **2** (zdecydowanie się zgadzam).
- "Jak bardzo zgadzasz się ze stwierdzeniem, że twoje wynagrodzenie adekwatnie odzwierciedla zakres wykonywanych przez ciebie obowiązków? zmienna **PYT_2** przyjmująca wartości: **-2** (zdecydowanie się nie zgadzam), **-1** (nie zgadzam się), **1** (zgadzam się), **2** (zdecydowanie się zgadzam).

Dodatkowo w ramach metryczki ankietowani zostali poproszeni o wskazanie swojego wieku - zmienna **WIEK** przyjmująca wartości numeryczne, oraz wskazanie płci - zmienna **PŁEĆ** przyjmująca wartość **Kobieta** lub **Mężczyzna**.
Kilka tygodni później przeprowadzono rewizję wynagrodzeń, w wyniku której część pracowników otrzymała podwyżki. Ankietowanych biorących udział w badaniu poproszono wówczas o ponowną odpowiedź na pytanie dotyczące zadowolenia z wynagrodzenia - zmienna **PYT_3**.

3.2 Zadanie 1.1

Wczytaj dane i przygotuj je do analizy. Zadbaj o odpowiednie typy zmiennych, zweryfikuj czy przyjmują wartości zgodne z powyższym opisem, zbadaj czy nie występują braki w danych.

```
# wczytanie danych
data <- read.csv("ankieta.csv", fileEncoding = "Latin1", sep=";", na=c("")
    )

# zmiana nazw na zgodne z poleceniem
colnames(data) <- c('DZIAŁ','STAŻ','CZY_KIER', 'PYT_1', 'PYT_2', 'PYT_3',
     'PŁEĆ', 'WIEK')

# zbadanie czy dane maja odpowiednie typy
types <- sapply(data, class)
types_df <- data.frame(Zmienna = names(types), Typ = types, row.names =
    NULL)

# stworzenie tabeli z typami poszczególnych zmiennych
kable(types_df,
      caption = "Typy zmiennych w kolumnach",
      align = "c",
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))

# pomocnicza zamiana na factor
data$PYT_1 <- factor(data$PYT_1, levels = c(-2, -1, 0, 1, 2),
                     labels = c("zdecydowanie się nie zgadzam", "nie
    zgadzam się", "nie mam zdania", "zgadzam się", "zdecydowanie się
    zgadzam"))
data$PYT_2 <- factor(data$PYT_2, levels = c(-2, -1, 1, 2),
                     labels = c("zdecydowanie się nie zgadzam", "nie
    zgadzam się", "zgadzam się", "zdecydowanie się zgadzam"))
data$PYT_3 <- factor(data$PYT_3, levels = c(-2, -1, 1, 2),
                     labels = c("zdecydowanie się nie zgadzam", "nie
    zgadzam się", "zgadzam się", "zdecydowanie się zgadzam"))
```

Otrzymałyśmy tabelę, w której podane są typy danych.

ZMIENNA	TYP
DZIAŁ	character
STAŻ	integer
CZY_KIER	character
PYT_1	integer
PYT_2	integer
PYT_3	integer
PŁEĆ	character
WIEK	integer

Tabela 1: Typy danych

Jak możemy zauważyć w tabeli 1, zmienne **DZIAŁ**, **CZY_KIER** i **PŁEĆ** są zmiennymi kategorycznymi, natomiast **STAŻ**, **PYT_1**, **PYT_2**, **PYT_3** oraz **WIEK** to zmienne liczbowe, co jest zgodne z naszymi przypuszczeniami.

```
# zbadanie występowania braku danych
colSums(is.na(data))
```

Jako wynik działania powyższego kodu otrzymałyśmy tabelę prezentującą liczbę braków danych dla konkretnych zmiennych.

DZIAŁ	STAŻ	CZY_KIER	PYT_1	PYT_2	PYT_3	PŁEĆ	WIEK
0	0	0	0	0	0	0	0

Tabela 2: Występowanie braków danych

Jak możemy zauważyć w tabeli 2 w naszych danych nie obserwujemy braków.

3.3 Zadanie 1.2

Utwórz zmienną **WIEK_KAT** przeprowadzając kategoryzację zmiennej **WIEK** korzystając z następujących przedziałów: do 35 lat, między 36 a 45 lat, między 46 a 55 lat, powyżej 55 lat.

```
# Podział wieku na odpowiednie grupy
df <- mutate(data, WIEK_KAT = cut(WIEK, breaks = c(0, 35, 45, 55, max(
    WIEK)),
                                  labels = c("0-35", "36-45", "46-55", "
    56+")))
View(df)
```

3.4 Zadanie 1.3

Sporządź tablice liczności dla zmiennych: **DZIAŁ**, **STAŻ**, **CZY_KIER**, **PŁEĆ**, **WIEK_KAT**.

```
generate_frequency_table <- function(df, variable) {

# Tworzenie tablicy liczności
  result <- table(df[[variable]])

# Wyświetlenie informacji o brakujących danych
```

```r
  kable(result,
        caption = paste("Tablica liczności dla zmiennej", variable),
        align = "c",
        col.names = c(variable, "Liczba wystąpień"),
        booktabs = TRUE) %>%
    kable_styling(latex_options = c("striped", "hold_position"))
}

# Wyświetlanie tablic
generate_frequency_table(df, "DZIAŁ")
generate_frequency_table(df, "STAŻ")
generate_frequency_table(df, "CZY_KIER")
generate_frequency_table(df, "PŁEĆ")
generate_frequency_table(df, "WIEK_KAT")
```

DZIAŁ	Liczba wystąpień
DK	98
DS	45
HR	31
IT	26

Tabela 3: Tablica liczności dla zmiennej DZIAŁ

STAŻ	Liczba wystąpień
1	41
2	140
3	19

Tabela 4: Tablica liczności dla zmiennej STAŻ

CZY_KIER	Liczba wystąpień
Nie	173
Tak	27

Tabela 5: Tablica liczności dla zmiennej CZY_KIER

PŁEĆ	Liczba wystąpień
K	71
M	129

Tabela 6: Tablica liczności dla zmiennej PŁEĆ

WIEK_KAT	Liczba wystąpień
0-35	26
36-45	104
45-55	45
56+	25

Tabela 7: Tablica liczności dla zmiennej WIEK_KAT

W tabelach 3 - 7 widoczne są liczby wystąpień poszczególnych opcji, dla zmiennych **DZIAŁ** (tabela 3), **STAŻ** (tabela 4), **CZY_KIER** (tabela 5), **PŁEĆ** (tabela 6) oraz **WIEK_KAT** (tabela 7).

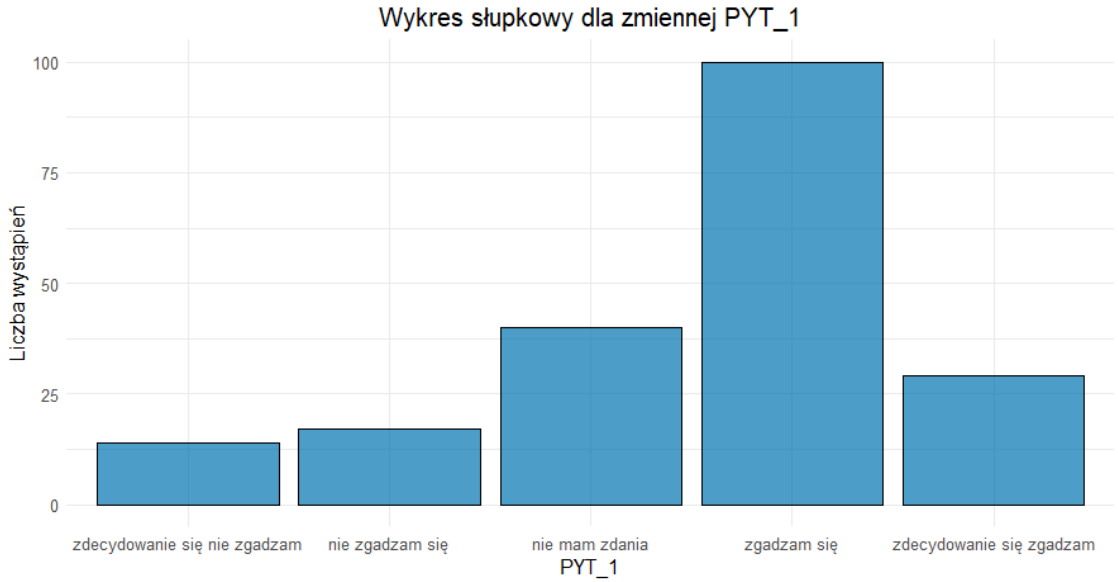
3.5 Zadanie 1.4

Sporządź wykresy kołowe oraz wykresy słupkowe dla zmiennych: **PYT_1** oraz **PYT_2**.

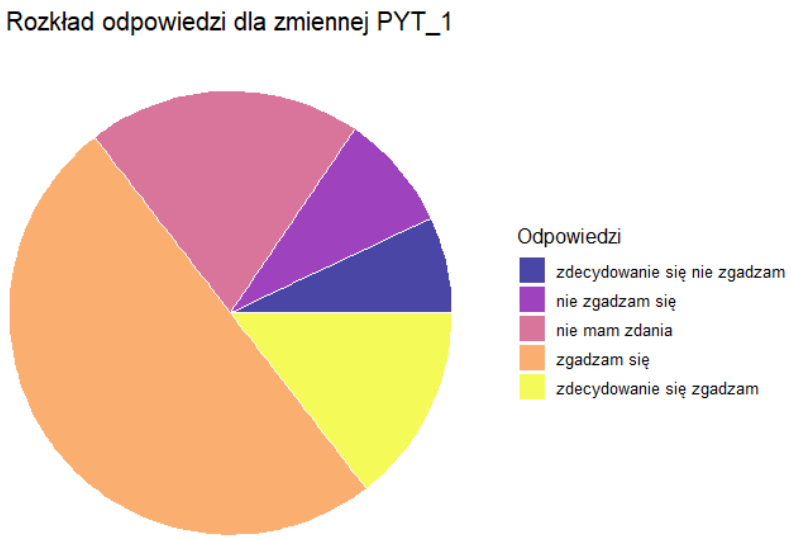
```
#------------------------------- PYT_1 ----------------------------------
# Wykres słupkowy dla pytania 1
ggplot(df, aes(x = PYT_1)) +
  geom_bar(fill = "#0072B2", color = "black", alpha = 0.7) +
  labs(title = "Wykres słupkowy dla zmiennej PYT_1", x = "PYT_1", y = "
   Liczba wystąpień") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))

# Wykres kołowy dla pytania 1
ggplot(data = df, aes(x = "", fill = PYT_1)) +
  geom_bar(color = "white", position="fill", alpha = 0.75) +
  coord_polar("y", start=pi/2) +
  scale_fill_viridis_d(option = "plasma", na.value = "gray50") +
  labs(title = "Rozkład odpowiedzi dla zmiennej PYT_1", fill = "
   Odpowiedzi") +
  theme_void() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15),
    axis.text = element_blank(),
    axis.title = element_blank(),
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10)
  )
```

```
#------------------------------- PYT_2 ----------------------------------
# Wykres słupkowy dla pytania 2
ggplot(df, aes(x = PYT_2)) +
  geom_bar(fill = "#0072B2", color = "black", alpha = 0.7) +
  labs(title = "Wykres słupkowy dla zmiennej PYT_2", x = "PYT_2", y = "
   Liczba wystąpień") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))

```

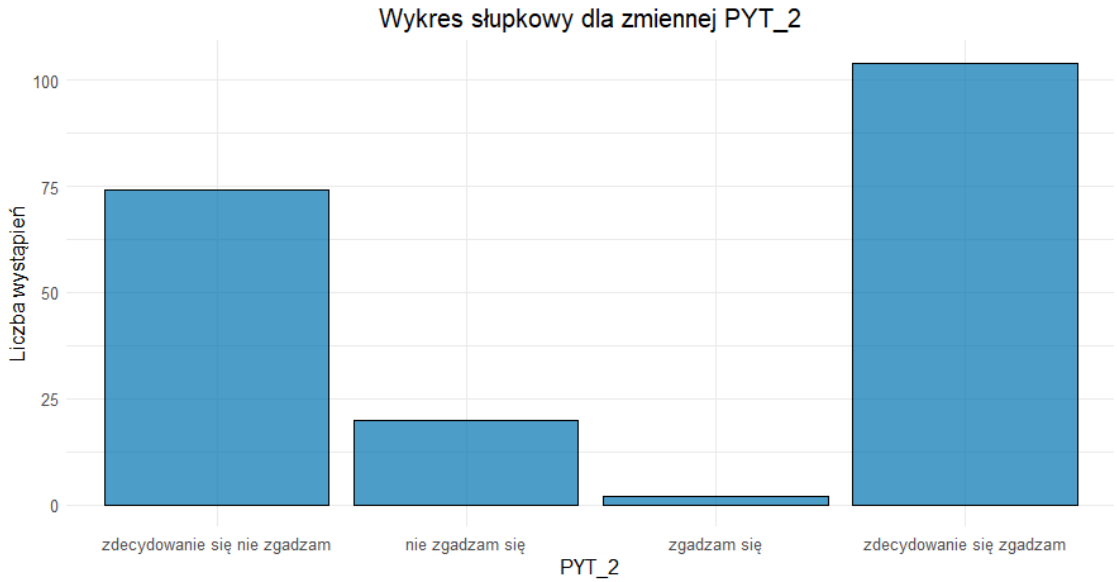
```r
# Wykres kołowy dla pytania 2
ggplot(data = df, aes(x = "", fill = PYT_2)) +
  geom_bar(color = "white", position="fill", alpha = 0.75) +
  coord_polar("y", start=pi/2) +
  scale_fill_viridis_d(option = "plasma", na.value = "gray50") +
  labs(title = "Rozkład odpowiedzi dla zmiennej PYT_2", fill = "
   Odpowiedzi") +
  theme_void() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 15),
    axis.text = element_blank(),
    axis.title = element_blank(),
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10)
  )
```



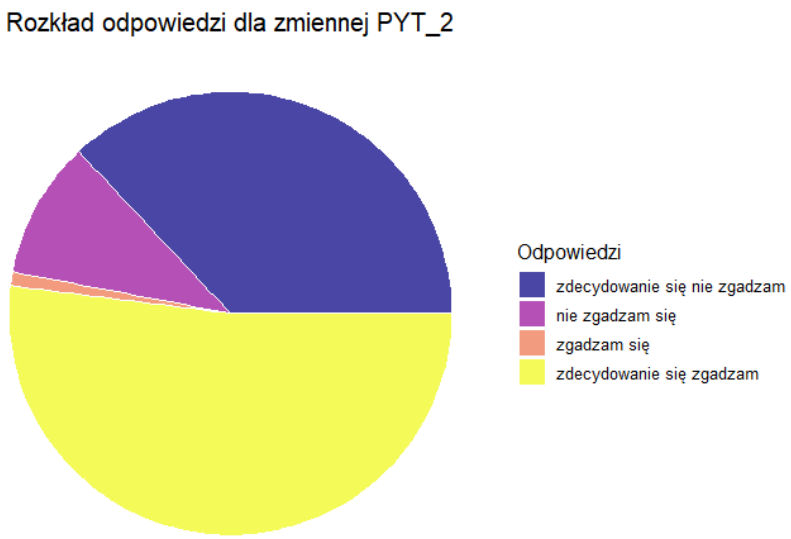
Rysunek 1: Wykres słupkowy dla zmiennej PYT_1



Rysunek 2: Wykres kołowy dla PYT_1



Rysunek 3: Wykres słupkowy dla PYT_2



Rysunek 4: Wykres kołowy dla PYT_2

Na rysunkach 1 oraz 3 zostały przedstawione wykresy słupkowe odpowiednio dla zmiennych **PYT_1** i **PYT_2**. Natomiast na rysunkach 2 oraz 4 wykresy kołowe dla tych samych zmiennych. Jak możemy zauważyć na wykresie 1 i 2, większość ankietowanych w odpowiedzi na pytanie 1 zaznaczyła opcję "zgadzam się". Natomiast najmniej osób zaznaczyło negatywne odpowiedzi. W przypadku pytania 2 (wykresy 3, 4) przeważają skrajne opcje. Najwięcej osób zaznaczyło odpowiedź "zdecydowanie się zgadzam", jednak prawie równie popularna jest odpowiedź "zdecydowanie się nie zgadzam".

3.6 Zadanie 1.5

Sporządź tablice wielodzielcze dla par zmiennych: **PYT_1** i **DZIAŁ**, **PYT_1** i **STAŻ**, **PYT_1** i **CZY_KIER**, **PYT_1** i **PŁEĆ** oraz **PYT_1** i **WIEK_KAT**.

```
generate_multiplication_table <- function(df, variable_1, variable_2) {
  # Tworzenie tablicy wielodzielczej
  result <- table(df[[variable_1]], df[[variable_2]])

  # Wyświetlenie tablic z odpowiednimi opcjami stylizacji
  kable(result,
        caption = paste("Tablica wielodzielcza dla zmiennych", variable_
    1, "i", variable_2),
        align = "c",
        booktabs = TRUE) %>%
    kable_styling(latex_options = c("striped", "hold_position"))
}

# Wyświetlanie tablic
generate_multiplication_table(df, "PYT_1", "DZIAŁ")
generate_multiplication_table(df, "PYT_1", "STAŻ")
generate_multiplication_table(df, "PYT_1", "CZY_KIER")
generate_multiplication_table(df, "PYT_1", "PŁEĆ")
generate_multiplication_table(df, "PYT_1", "WIEK_KAT")
```

W wyniku działania powyższego kodu otrzymaliśmy 5 tablic wielodzielczych, które są widoczne poniżej.

	DK	**DS**	**HR**	**IT**
zdecydowanie się nie zgadzam	9	3	2	0
nie zgadzam się	10	3	2	2
nie mam zdania	17	14	5	4
zgadzam się	51	15	19	15
zdecydowanie się zgadzam	11	10	3	5

Tabela 8: Tablica wielodzielcza dla zmiennych PYT_1 i DZIAŁ

	1	**2**	**3**
zdecydowanie się nie zgadzam	5	5	4
nie zgadzam się	6	10	1
nie mam zdania	8	26	6
zgadzam się	19	75	6
zdecydowanie się zgadzam	3	24	2

Tabela 9: Tablica wielodzielcza dla zmiennych PYT_1 i STAŻ

	Nie	**Tak**
zdecydowanie się nie zgadzam	10	4
nie zgadzam się	14	3
nie mam zdania	34	6
zgadzam się	88	12
zdecydowanie się zgadzam	27	2

Tabela 10: Tablica wielodzielcza dla zmiennych PYT_1 i CZY_KIER

	K	**M**
zdecydowanie się nie zgadzam	3	11
nie zgadzam się	7	10
nie mam zdania	14	26
zgadzam się	36	64
zdecydowanie się zgadzam	11	18

Tabela 11: Tablica wielodzielcza dla zmiennych PYT_1 i PŁEĆ

	0-35	36-45	46-55	56+
zdecydowanie się nie zgadzam	1	11	2	0
nie zgadzam się	6	7	1	3
nie mam zdania	3	24	5	8
zgadzam się	13	50	25	12
zdecydowanie się zgadzam	3	12	12	2

Tabela 12: Tablica wielodzielcza dla zmiennych PYT_1 i WIEK_KAT

W tabelach 8 - 12 widoczne są tablice wielodzielcze odpowiednio dla par zmiennych:

- **PYT_1** i **DZIAŁ** - tabela 8,
- **PYT_1** i **STAŻ** - tabela 9,
- **PYT_1** i **CZY_KIER** - tabela 10,
- **PYT_1** i **PŁEĆ** - tabela 11,
- **PYT_1** i **WIEK_KAT** - tabela 12.

Możemy z nich wyczytać liczbę wystąpień dla poszczególnych par z danej kategorii.

3.7 Zadanie 1.6

Sporządź tablicę wielodzielczą dla pary zmiennych: **PYT_2** i **PYT_3**.

```
# zwykła tablica wielodzielcza
generate_multiplication_table(df, "PYT_2", "PYT_3")

# tablica wielodzielcza z sumami
generate_multiplication_table_2 <- function(df, variable_1, variable_2) {

  # Tworzenie tablicy licznosci
  result <- table(df[[variable_1]], df[[variable_2]])

  # Policzenie sumy dla każdego wiersza
  row_sum <- rowSums(result)

  # Policzenie sumy dla każdej kolumny
  col_sum  <- colSums(result)

  # Dodanie sum do tabeli jako dodatkowych wierszy i kolumn
  sum_table <- cbind(result, "Suma" = row_sum)
  sum_table <- rbind(sum_table, "Suma" = c(col_sum, ""))

  # Zwrócenie tabeli
  kable(sum_table,
        caption = paste("Tabela wielodzielcza z sumami dla zmiennych",
    variable_1, "i", variable_2),
        align = "c",
        booktabs = TRUE) %>%
    kable_styling(latex_options = c("striped", "hold_position"))
}

generate_multiplication_table_2(df, "PYT_2", "PYT_3")
```

W wyniku działania powyższego kodu otrzymaliśmy tablicę wielodzielczą z sumami dla zmiennych **PYT_2** i **PYT_3**, która jest widoczna poniżej.

	zdecydowanie się nie zgadzam	**nie zgadzam się**	**zgadzam się**	**zdecydowanie się zgadzam**	**Suma**
zdecydowanie się nie zgadzam	49	16	5	4	74
nie zgadzam się	3	6	10	1	20
zgadzam się	0	0	2	0	2
zdecydowanie się zgadzam	0	8	15	81	104
Suma	52	30	32	86	

Tabela 13: Tablica wielodzielcza z sumami dla zmiennych PYT_2 i PYT_3

W tabeli 13 widzimy jak wygląda liczba wystąpień dla poszczególnych par odpowiedzi na **PYT_2** i **PYT_3**.

3.8 Zadanie 1.7

Utwórz zmienną **CZY_ZADOW** na podstawie zmiennej **PYT_2** łącząc kategorie "nie zgadzam się"i "zdecydowanie się nie zgadzam"oraz "zgadzam się"i "zdecydowanie się zgadzam".

Wybraliśmy nowszą wersję zadania, czyli tą, w której kategorie tworzymy na podstawie zmiennej **PYT_2**. Poniżej widoczny jest kod, który wykorzystałyśmy do tego celu.

```
# Tworzenie zmiennej CZY_ZADOW na podstawie zmiennej PYT_2
df$CZY_ZADOW <- ifelse(df$PYT_2 %in% c("zdecydowanie się nie zgadzam", "
    nie zgadzam się"), "NIE", "TAK")
```

Powyższy kod tworzy nową zmienną **CZY_ZADOW**, którą wykorzystamy w późniejszych zadaniach.

3.9 Zadanie 1.8

Korzystając z funkcji *mosaic* z biblioteki *vcd*, sporządź wykresy mozaikowe odpowiadające parom zmiennych: **CZY_ZADOW** i **DZIAŁ**, **CZY_ZADOW** i **STAŻ**, **CZY_ZADOW** i **CZY_KIER**, **CZY_ZADOW** i **PŁEĆ** oraz **CZY_ZADOW** i **WIEK_KAT**. Czy na podstawie uzyskanych wykresów można postawić pewne hipotezy dotyczące realizacji między powyższymi zmiennymi? Spróbuj sformułować kilka takich hipotez.

W tym zadaniu wykorzystałyśmy funkcję *mosaic* z biblioteki *vcd*. Funkcja ta służy do tworzenia wykresów mozaikowych, które mają zastosowanie w wizualizacji zależności pomiędzy dwoma zmiennymi kategorycznymi.

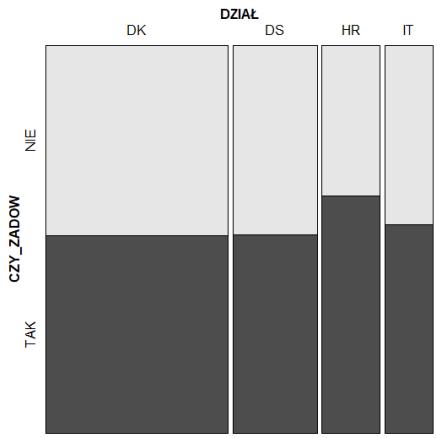
Na podstawie danych wejściowych funkcja *mosaic* tworzy prostokątne obszary (mozaikę), które są proporcjonalne do liczby obserwacji w każdej kombinacji kategorii.

Tego typu wykresy pozwalają nam łatwo zauważyć, które kombinacje kategorii są bardziej lub mniej powszechne, co może pomóc nam dostarczyć informacji na temat zależności między poszczególnymi zmiennymi.

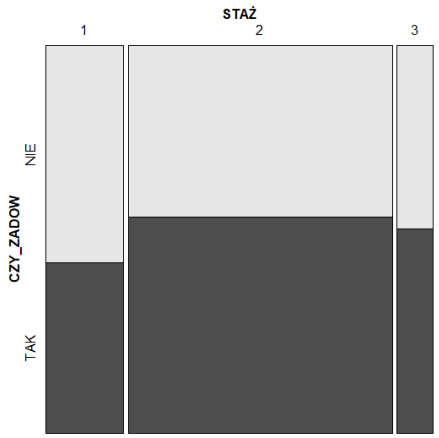
```
mosaic( ~ CZY_ZADOW + DZIAŁ, data = df, highlight = TRUE)
mosaic( ~ CZY_ZADOW + STAŻ, data = df, highlight = TRUE)
mosaic( ~ CZY_ZADOW + CZY_KIER, data = df, highlight = TRUE)
mosaic( ~ CZY_ZADOW + PŁEĆ, data = df, highlight = TRUE)
mosaic( ~ CZY_ZADOW + WIEK_KAT, data = df, highlight = TRUE)
```

Korzystając z powyższego kodu sporządziłyśmy 5 wykresów mozaikowych dla odpowiednich par zmiennych:

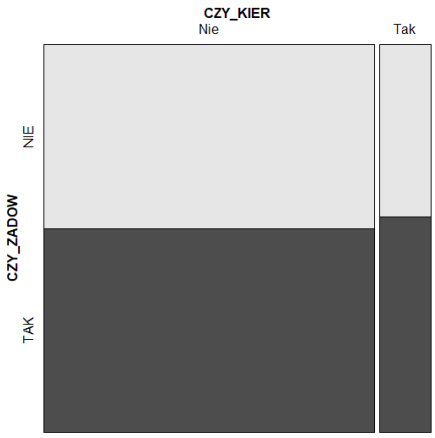
- **CZY_ZADOW** i **DZIAŁ** - rysunek 2,
- **CZY_ZADOW** i **STAŻ** - rysunek 3,
- **CZY_ZADOW** i **CZY_KIER** - rysunek 4,
- **CZY_ZADOW** i **PŁEĆ** - rysunek 5,
- **CZY_ZADOW** i **WIEK_KAT** - rysunek 6.



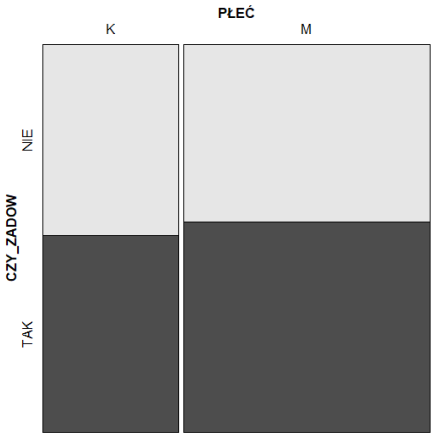
Rysunek 5: CZY_ZADOW i DZIAŁ



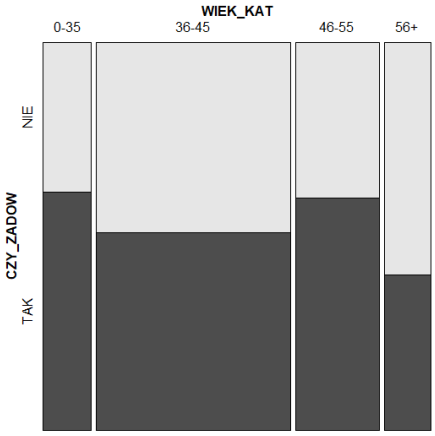
Rysunek 6: CZY_ZADOW i STAŻ



Rysunek 7: CZY_ZADOW i CZY_KIER



Rysunek 8: CZY_ZADOW i PŁEĆ



Rysunek 9: CZY_ZADOW i WIEK_KAT

Analizując powyższe wykresy możemy wyciągnąć kilka hipotez, między innymi:

1. Istnieje zależność pomiędzy poziomem zadowolenia pracowników, a działem w jakim oni pracują.
2. Zazwyczaj pracownicy z dłuższym stażem mają tendencję do bycia bardziej zadowolonymi niż ci ze stażem krótszym.
3. Pracownicy zajmujący kierownicze stanowiska są bardziej zadowoleni niż ci na niższych stanowiskach.
4. Płeć pracowników może mieć wpływ na ich zadowolenie z pracy.
5. Istnieje dość znaczący związek pomiędzy wiekiem pracowników, a ich poziomem zadowolenia z pracy.

Możemy także zauważyć, że najwięcej osób pracuje w dziale kreatywnym (DK), przeważają osoby w wieku 36-45, większość osób nie pracuje na stanowisku kierowniczym. Dodatkowo widzimy, że w firmie pracuje więcej mężczyzn.

4 Część II

4.1 Zadanie 2

Zapoznaj się z biblioteką *likert* i dostępnymi tam funkcjami *summary* oraz *plot* (wykresy typu "bar", "heat"oraz "density"), a następnie zilustruj odpowiedzi na pytanie "Jak bardzo zgadzasz się ze stwierdzeniem, że firma pozwala na (...)?"(zmienna **PYT_1**) w całej badanej grupie oraz w podgrupach ze względu na zmienną **CZY_KIER**.

Do tego zadania wykorzystałyśmy funkcję *summary* oraz *plot* z biblioteki *likert*.

Funkcja *summary* generuje podsumowanie danych związane ze skalami Likerta, w formie tabeli zawierającej następujące kolumny:

- Item - pytanie, dla którego są prezentowane wyniki,
- low - ankietowani, którzy odpowiedzieli na zadane pytanie na niższym końcu skali: "zdecydowanie się nie zgadzam"lub "nie zgadzam się",
- neutral - ankietowani, którzy odpowiedzieli neutralnie na zadane pytanie "nie mam zdania",
- high - ankietowani, którzy odpowiedzieli na zadane pytanie na wyższym końcu skali: "zgadzam się"lub "zdecydowanie się zgadzam",
- mean - wartość średnia,
- sd - odchylenie standardowe z odpowiedzi.

Natomiast funkcja *plot* służy do wizualizacji tych danych. Za jej pomocą możemy stworzyć między innymi:

- wykres słupkowy (bar plot),
- wykres rozkładu gęstości (density plot),
- wykres mapy ciepła/macierzy kolorów (heatmap).

W pierwszej części tego zadania zilustrowałyśmy odpowiedź na **PYT_1** w całej badanej grupie za pomocą wykresów typu "bar", "heat"oraz "density". W tym celu wykorzystałyśmy kod widoczny poniżej.

```
df$CZY_KIER <- factor(df$CZY_KIER, levels = c("Nie", "Tak"))
head(df)

#_____________________________ bez grupowania ______________________________
# Przekształcenie podsumowania na ramkę danych
likert_pyt_1 <- likert(df[,"PYT_1", drop=FALSE])

summ_pyt_1 <- summary(likert_pyt_1)

```

```r
# Wyświetlenie podsumowania w formie tabeli
kable(summ_pyt_1,
      caption = "Podsumowanie odpowiedzi na pytanie PYT_1",
      align = "c",
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

W wyniku działania funkcji *summary* otrzymałyśmy poniższą tabelę.

Item	low	neutral	high	mean	sd
PYT_1	15.5	20	64.5	3.565	1.063688

Tabela 14: Podsumowanie odpowiedzi na pytanie PYT_1 bez grupowania

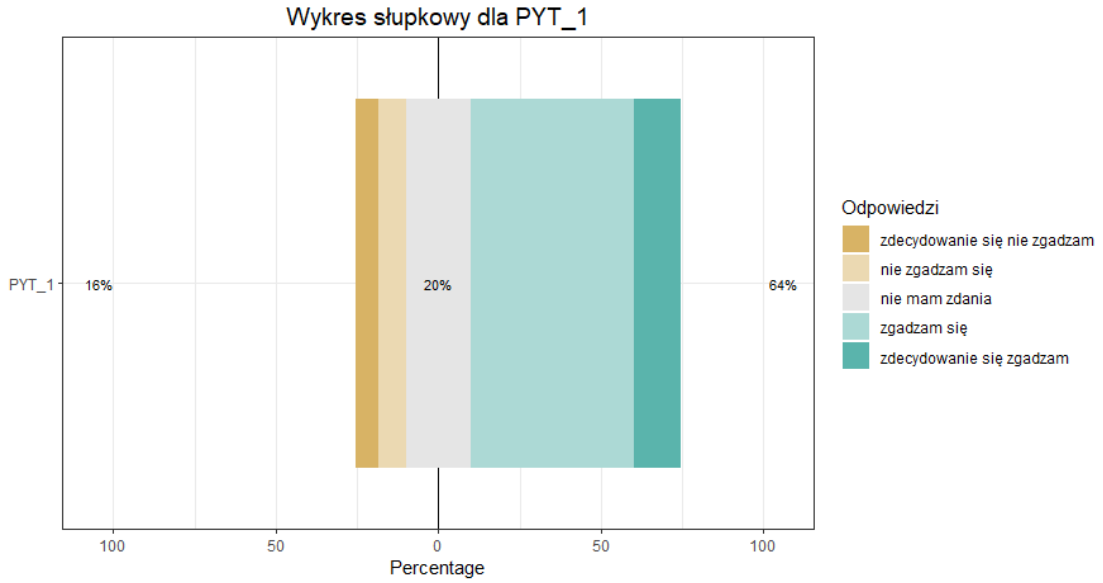
Jak możemy zauważyć w tabeli 14, większość osób (64.5%) uważa, że firma pozwala na elastyczne godziny pracy, 20% osób nie ma na ten temat zdania, natomiast pozostałe 15.5% uważa, że firma nie umożliwia zachowania równowagi pomiędzy pracą, a życiem prywatnym. Warto również zaznaczyć, że wyniki zostały opracowane na podstawie 5-stopniowej skali, gdzie:

- 1 - zdecydowanie się nie zgadzam,
- 2 - nie zgadzam się,
- 3 - nie mam zdania,
- 4 - zgadzam się,
- 5 - zdecydowanie się zgadzam.

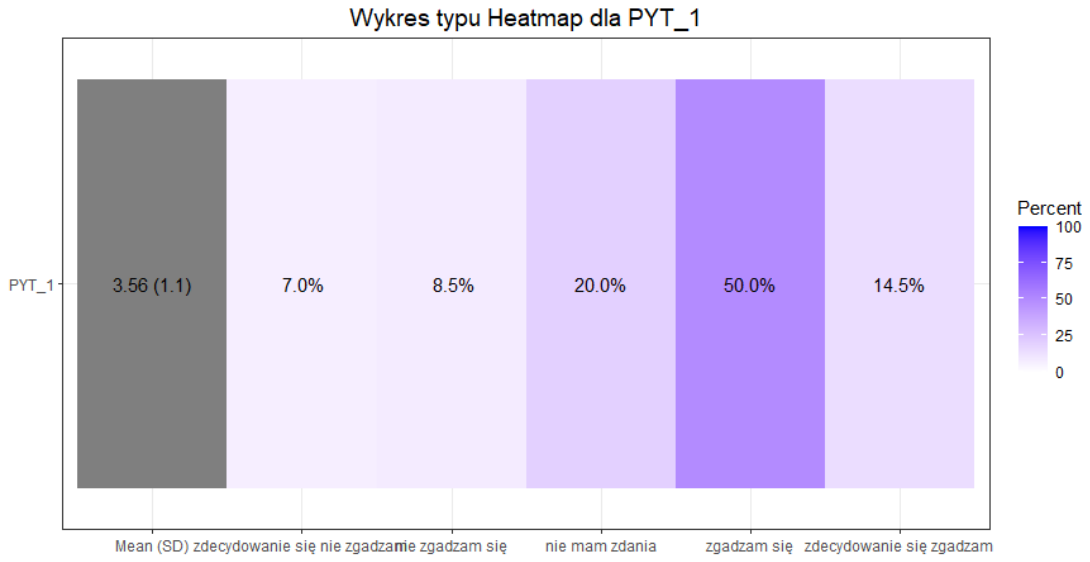
Widzimy, że średnia wynosi 3.565 co pozwala nam stwierdzić, że większość osób uważa, że firma pozwala na elastyczne godziny pracy. Odchylenie standardowe jest równe 1.063688 co pozwala stwierdzić, że niewiele osób zaznaczyło skrajne odpowiedzi.

Następnie, z pomocą funkcji *plot* sporządziłyśmy wykres słupkowy, mapy ciepła oraz gęstości.

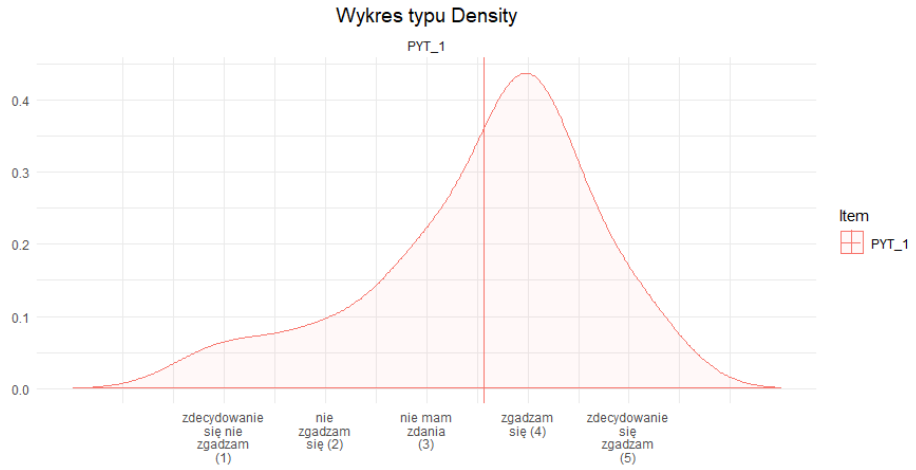
```r
plot(likert_pyt_1, type = "bar", auto.key = list(columns = 2)) +
  labs(title = "Wykres słupkowy dla PYT_1") +
  guides(fill = guide_legend(title = "Odpowiedzi")) +
  theme_bw() +
  theme(plot.title = element_text(size = 14,, hjust = 0.5))

plot(likert_pyt_1, type = "heat") +
  labs(title = "Wykres typu Heatmap dla PYT_1") +
  theme_bw() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))

plot(likert_pyt_1, type = "density") +
  theme_minimal() +
  labs(title = "Wykres typu Density") +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```



Rysunek 10: Wykres słupkowy dla zmiennej PYT_1



Rysunek 11: Wykres typu Heatmap dla zmiennej PYT_1



Rysunek 12: Wykres typu Density dla zmiennej PYT_1

Na rysunkach 10 - 12 możemy zauważyć, że nasza teza mówiąca o tym, że większość osób pracujących w firmie uważa, że oferuje ona elastyczny grafik pracy, jest prawdziwa. Widzimy także, że największy odsetek osób zaznaczyło opcję "zgadzam się" na wykresie 10 pole to jest najszersze, na wykresie 11 kolor jest najciemniejszy, natomiast na wykresie 12 w tym miejscu występuje pik funkcji gęstości. Na wszystkich powyższych wykresach widzimy także, że najmniej osób zaznaczyło odpowiedź "zdecydowanie się nie zgadzam".

Następnie pogrupowałyśmy ankietowanych ze względu na fakt zajmowania przez nich kierowniczych stanowisk (zmienna **CZY_KIER**) i zwizualizowałyśmy wyniki na wykresie typu "bar".

```
#_____________________________ z grupowaniem _____________________________
likert_pyt_1_grouped <- likert(df[, "PYT_1", drop = FALSE], grouping = df
    $CZY_KIER)

summ_pyt_1_grouped <- summary(likert_pyt_1_grouped)
print(summ_pyt_1_grouped)

# Wyświetlenie podsumowania w formie tabeli
kable(summ_pyt_1_grouped,
      caption = "Podsumowanie odpowiedzi na pytanie PYT_1",
      align = "c",
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Group	Item	low	neutral	high	mean	sd
Nie	PYT_1	13.87283	19.65318	66.47399	3.624277	1.030291
Tak	PYT_1	25.92593	22.22222	51.85185	3.185185	1.210119

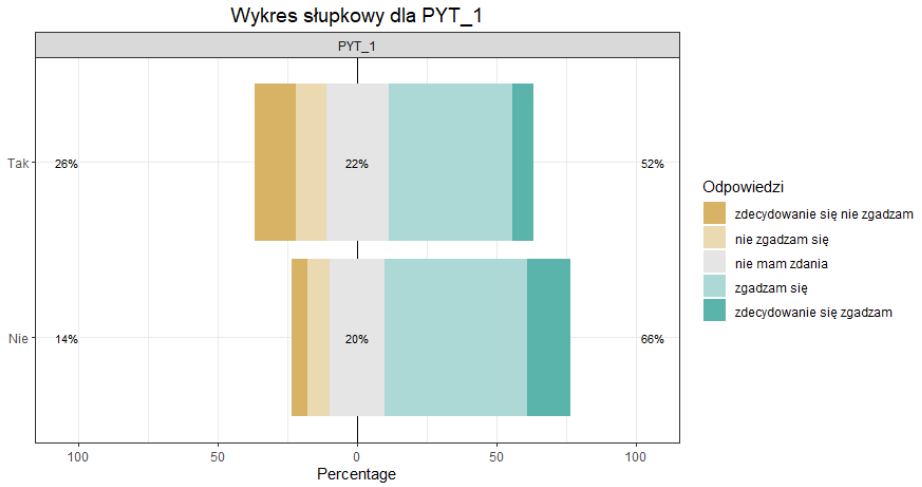
Tabela 15: Podsumowanie odpowiedzi na pytanie PYT_1 z grupowaniem

Jak możemy zauważyć w tabeli 15, także sporządzonej z pomocą funkcji *summary*, osoby pracujące na stanowisku kierowniczym w większym stopniu nie zgadzają się ze stwierdzeniem, że firma umożliwia zachowanie równowagi pomiędzy pracą i życiem prywatnym

(jednak ciągle są zadowoleni z elastyczności grafiku). W tym przypadku, podobnie jak w zadaniu 2 została użyta 5-stopniowa skala (4.1). Analizując zachowanie średniej potwierdza się nasza teza mówiąca o tym, że osoby nie pracujące na stanowisku kierowniczym w większym stopniu zgadzają się ze stwierdzeniem, że firma oferuje elastyczny grafik pracy. Patrząc na odchylenie standardowe widzimy, że wśród osób zajmujących kierownicze stanowiska odpowiedzi są bardziej zróżnicowane (jednak z przewagą odpowiedzi pozytywnych).

W tym przypadku także podjęłyśmy próbę zwizualizowania odpowiedzi na wykresach.

```
plot(likert_pyt_1_grouped, type = "bar", auto.key = list(columns = 2)) +
  labs(title = "Wykres słupkowy dla PYT_1") +
  guides(fill = guide_legend(title = "Odpowiedzi")) +
  theme_bw() +
  theme(plot.title = element_text(size = 14, hjust = 0.5))
```



Rysunek 13: Wykres słupkowy dla zmiennej PYT_2

W tym przypadku udało nam się sporządzić wykres słupkowy (bar plot), który widoczny jest na rysunku 13. Analizując jego wygląd możemy zauważyć, że nasza teza potwierdziła się. W obu podgrupach zdecydowanie przeważa odpowiedź "zgadzam się"(najszerszy przedział). W zależności od zajmowanego stanowiska widoczna jest dość znacząca różnica w skrajnych odpowiedziach: "zdecydowanie się zgadzam"i "zdecydowanie się nie zgadzam". Widzimy także, że osoby zaznaczające odpowiedź "nie mam zdania"stanowią w obu podgrupach podobny odsetek. Podsumowując, możemy stwierdzić, że osoby nie zajmujące kierowniczych stanowisk w większej części zgadzają się ze stwierdzeniem, że firma zapewnia elastyczny grafik pracy. Jednak wszyscy, bez względu na zajmowane stanowisko, są raczej zadowoleni z elastyczności grafiku.

4.2 Zadanie 3

Zapoznaj się z funkcją *sample* z biblioteki *stats*, a następnie wylosuj próbkę o liczności 10% wszystkich rekordów z pliku "ankieta.csv"w dwóch wersjach: ze zwracaniem oraz bez zwracania.

W tym zadaniu wykorzystałyśmy funkcję *sample* z biblioteki *stats*. Służy ona do losowego próbkowania elementów z zestawu danych. Pozwala na wybieranie losowych próbek danych bez zwracania i ze zwracaniem.

```r
# Liczność próbki (10% wszystkich rekordów)
sample_size <- round(0.1 * nrow(df))

# losowanie ze zwracaniem
with_returning <- sample(1:nrow(df), size = sample_size, replace = TRUE)

# losowanie bez zwracania
without_returning <- sample(1:nrow(df), size = sample_size, replace =
    FALSE)

# Wyświetlenie wylosowanych próbek
kable(with_returning,
      caption = "Wylosowana próbka ze zwracaniem",
      align = "c",
      col.names = c("Próbka"),
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))

kable(without_returning,
      caption = "Wylosowana próbka bez zwracania",
      align = "c",
      col.names = c("Próbka"),
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

W poniższej tabeli przedstawiłyśmy wyniki działania powyższego kodu. W obu przypadkach wylosowałyśmy 20 rekordów, co stanowi 10% wszystkich obserwacji. Warto zaznaczyć, że są to jedynie przykładowe wyniki. Ze względu na fakt, że losowanie odbywa się w sposób losowy wyniki w przypadku każdorazowego wywołania kodu będą inne.

Losowanie ze zwracaniem	**Losowanie bez zwracania**
80	65
43	200
157	53
128	167
29	1
154	36
88	58
82	125
159	130
155	96
24	180
66	162
77	144
108	50
45	145
18	83
155	6
73	179
52	177
97	40

Tabela 16: Przykładowe wylosowane próbki ze zwracaniem i bez zwracania

W tabeli 16 widzimy przykładowe wyniki losowania ze zwracaniem i bez.

4.3 Zadanie 4

Zaproponuj metodę symulowania zmiennych losowych z rozkładu dwumianowego. Napisz funkcję do generowania realizacji, a następnie zaprezentuj jej działanie porównując wybrane teoretyczne i empiryczne charakterystyki dla przykładowych wartości parametrów rozkładu: n i p.

W celu wygenerowania zmiennych losowych z rozkładu dwumianowego wykorzystałyśmy poniższy algorytm:

1. Inicjalizacja wyników - tworzymy pusty wektor do przechowywania późniejszych wyników.
2. Dla każdej próbki:
   (a) Inicjalizujemy zmienną służącą do przechowywania liczby sukcesów w danej próbce.
   (b) Dla każdej z prób:
      i. Generujemy losową liczbę z rozkładu jednostajnego $U(0, 1)$.
      ii. Sprawdzamy czy liczba ta jest mniejsza niż zadane prawdopodobieństwo sukcesu p. Jeśli tak, to zwiększamy licznik sukcesów.
   (c) Zapisujemy liczbę sukcesów do wektora wynikowego.
3. Zwracamy wektor wynikowy.

Poniżej widoczny jest kod obrazujący działanie powyższego algorytmu.

```
generate_binomial <- function(n, p, size) {
  result <- numeric(size)
  for (i in 1:size) {
    successes <- 0
    for (j in 1:n) {
      if (runif(1) < p) {
        successes <- successes + 1
      }
    }
    result[i] <- successes
  }
  return(result)
}
```

W celu sprawdzenia poprawności działania zaproponowanej metody porównałyśmy ze sobą teoretyczne i empiryczne wartości średniej i wariancji. Narysowałyśmy także teoretyczne i empiryczne rozkłady prawdopodobieństw oraz dystrybuant. Zarówno liczba próbek ("size"), jak i liczba prób ("n") wynosiła 1000. Wartości prawdopodobieństw sukcesu p ("p-values") należą do przedziału $0.1, 0.5, 0.9$.

```
n <- 1000
p_values <- c(0.1, 0.5, 0.9)
size <- 1000

empirical_p <- numeric(length(p_values))
```

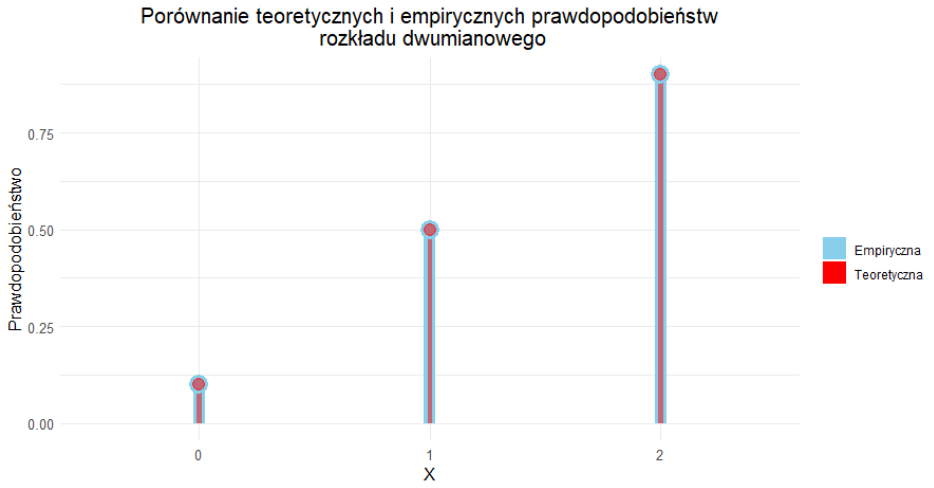
```r

theoretical_mean <- numeric(length(p_values))
theoretical_variance <- numeric(length(p_values))
empirical_mean <- numeric(length(p_values))
empirical_variance <- numeric(length(p_values))

for (i in 1:length(p_values)) {
  p <- p_values[i]

  empirical_binomial <- generate_binomial(n, p, size)
  empirical_p[i] <- mean(empirical_binomial) / n

  theoretical_mean[i] <- n * p
  theoretical_variance[i] <- n * p * (1 - p)

  empirical_mean[i] <- mean(empirical_binomial)
  empirical_variance[i] <- var(empirical_binomial)

}

results_binomial <- data.frame(
  "Charakterystyka" = c("Teoretyczna średnia", "Empiryczna średnia", "
   Teoretyczna wariancja", "Empiryczna wariancja"),
  "p = 0.1" = c(theoretical_mean[1], empirical_mean[1], theoretical_
   variance[1], empirical_variance[1]),
  "p = 0.5" = c(theoretical_mean[2], empirical_mean[2], theoretical_
   variance[2], empirical_variance[2]),
  "p = 0.9" = c(theoretical_mean[3], empirical_mean[3], theoretical_
   variance[3], empirical_variance[3])
)


kable(results_binomial,
      caption = "Porównanie empirycznych statystyk z teoretycznymi",
      align = "c",
      col.names = c("Charakterystyka", "p = 0.1", "p = 0.5", "p = 0.9"),
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Charakterystyka	$p = 0.1$	$p = 0.5$	$p = 0.9$
Teoretyczna średnia	100.0000	500.0000	900.00000
Empiryczna średnia	99.5460	499.3590	899.52600
Teoretyczna wariancja	90.0000	250.0000	90.00000
Empiryczna wariancja	88.4143	259.1272	92.85418

Tabela 17: Porównanie teoretycznych i empirycznych charakterystyk

W tabeli numer 17 zaprezentowano przykładowe wyniki wysymulowania zmiennych z rozkładu dwumianowego $B(1000, 0.1)$, $B(1000, 0.5)$, $B(1000, 0.9)$. Dla tych rozkładów porównałyśmy teoretyczne i empiryczne średnie oraz wariancje. Jak możemy zauważyć, są one do siebie zbliżone co pozwala nam wnioskować, że zaproponowana przez nas metoda jest najprawdopodobniej poprawna. W celu zweryfikownaia tej tezy sporządziłyśmy wykresy teoretycznych i empirycznych rozkładów prawdopodobieństwa oraz dystrybuant dla każdego p. Poniżej widoczny jest kod oraz wyniki.

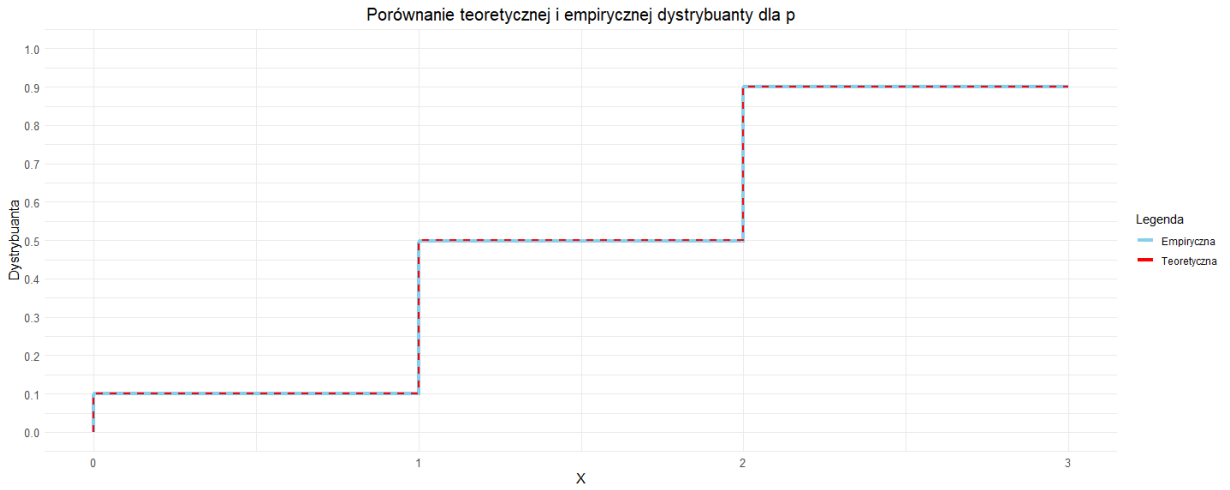
```r
results_comparison <- data.frame(
  "Teoretyczne" = p_values,
  "Empiryczne" = empirical_p
)
ggplot(results_comparison, aes(x = factor(0:(length(Teoretyczne) - 1)))) +
  geom_bar(aes(y = Empiryczne, fill = "Empiryczna"), stat = "identity", width = 0.05) +
  geom_bar(aes(y = Teoretyczne, fill = "Teoretyczna"), stat = "identity", width = 0.02, alpha = 0.5) +
  geom_point(aes(y = Empiryczne, color = "Empiryczna"), size = 6) +
  geom_point(aes(y = Teoretyczne, color = "Teoretyczna"), size = 4, alpha = 0.5) +
  scale_fill_manual(values = c("Teoretyczna" = "red", "Empiryczna" = "skyblue"), name = "") +
  scale_color_manual(values = c("Teoretyczna" = "red", "Empiryczna" = "skyblue"), name = "") +
  labs(x = "X", y = "Prawdopodobieństwo") +
  ggtitle("Porównanie teoretycznych i empirycznych prawdopodobieństw\n rozkładu dwumianowego") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))
```



Rysunek 14: Porównanie teoretycznych i empirycznych prawdopodobieństw

```r
sorted_empirical <- sort(empirical_p)

ggplot() +
  geom_step(data = data.frame(x = c(0, 0, 1, 2, 3), y = c(0, sorted_empirical, max(sorted_empirical))), aes(x, y),
            color = "skyblue", linetype = "solid", size = 1.5) +
  geom_step(data = data.frame(x = c(0, 0, 1, 2, 3), y = c(0, p_values, max(sorted_empirical))), aes(x, y),
            color = "red", linetype = "dashed", size = 1) +
  labs(x = "X", y = "Dystrybuanta",
       title = "Porównanie teoretycznej i empirycznej dystrybuanty dla p"
  ) +
```

```
  scale_x_continuous(limits = c(0, 3), breaks = c(0, 1, 2, 3)) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0, 1, by = 0.1)) +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))
```



Rysunek 15: Porównanie dystrybuanty teoretycznej i empirycznej

Jak możemy wnioskować z wykresu 14 (porównanie teoretycznych i empirycznych prawdopodobieństw) oraz z wykresu 15 (porównanie teoretycznej i empirycznej dystrybuanty) zaproponowana przez nas metoda symulacji najprawdopodobniej jest poprawna. Widzimy, że teoretyczne i empiryczne wartości się dość dobrze pokrywają dla każdej z badanych wartości prawdopodobieństwa sukcesu p.

4.4 Zadanie 5

Zaproponuj metodę symulowania wektorów losowych z rozkładu wielomianowego. Napisz funkcję do generowania realizacji, a następnie zaprezentuj jej działanie porównując wybrane teoretyczne i empiryczne charakterystyki dla przykładowych wartości parametrów rozkładu: n i p.

W celu wygenerowania wektorów losowych z rozkładu wielowymiarowego wykorzystałyśmy poniższy algorytm:

1. Inicjalizacja wyników - tworzymy pusty wektor do przechowywania późniejszych wyników.

2. Inicjalizacja pomocniczych zmiennych określających pozostałą liczbę prób oraz sumującą prawdopodobieństwa p.

3. Dla każdego p (z wyjątkiem ostatniego):

   (a) Jeśli suma prawdopodobieństw jest różna od 0 to generujemy losową zmienną z rozkładu dwumianowego.

(b) Aktualizujemy liczbę prób pozostałych do wykonania oraz sumę prawdopodobieństw.

4. Do ostatniej próbki przypisujemy pozostałe próby.

5. Zwracamy wektor wynikowy zawierający liczbę sukcesów dla każdej próbki.

Poniżej widoczny jest kod obrazujący działanie powyższego algorytmu.

```
generate_multinomial <- function(n, p) {
  result <- numeric(length(p))
  remaining_trials <- n
  prob_sum <- 1

  for (i in 1:(length(p) - 1)) {
    if (prob_sum != 0) {
      result[i] <- rbinom(1, remaining_trials, p[i]/prob_sum)
    } else {
      result[i] <- 0
    }
    remaining_trials <- remaining_trials - result[i]
    prob_sum <- prob_sum - p[i]
  }

  result[length(p)] <- remaining_trials
  return(result)
}
```

W celu sprawdzenia poprawności działania zaproponowanej metody porównałyśmy ze sobą teoretyczne i empiryczne wartości prawdopodobieństwa sukcesów p. Liczba prób wynosiła 10000. Wartości prawdopodobieństw sukcesu p wynoszą odpowiednio: $(0.1, 0.3, 0.4, 0.2)$, $(0.5, 0.3, 0.2)$, $(0.1, 0.1, 0.3, 0.2, 0.3)$.

```
p_values <- list(
  c(0.1, 0.3, 0.4, 0.2),
  c(0.5, 0.3, 0.2),
  c(0.1, 0.1, 0.3, 0.2, 0.3)
)

n <- 10000

results_df <- data.frame()


for (p in p_values) {
  sample <- generate_multinomial(n, p)

  empirical_p <- sample / n

  result <- data.frame(
    Theoretical_p = as.character(p),
    Empirical_p = as.character(empirical_p)
  )

  results_df <- rbind(results_df, result)
}

kable(results_df,
      caption = "Wylosowana próbka bez zwracania",
```

```
      align = "c",
      col.names = c("Teoretyczne p", "Empiryczne p"),
      booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Teoretyczne p	**Empiryczne** p
0.1	0.1056
0.3	0.3037
0.4	0.3914
0.2	0.1993
0.5	0.499
0.3	0.2993
0.2	0.2017
0.1	0.1002
0.1	0.1015
0.3	0.2994
0.2	0.1979
0.3	0.3010

Tabela 18: Porównanie teoretycznych i empirycznych wartości p

Jak możemy zauważyć w tabeli 18, wszystkie teoretyczne i empiryczne wartości prawdopodobieństw sukcesu p są do siebie zbliżone. Pozwala nam to wnioskować, że zaproponowana przez nas metoda symulacji najprawdopodobniej jest poprawna.

5 Część III i IV

5.1 Zadanie 6

Napisz funkcję do wyznaczania realizacji przedziału ufności Cloppera-Pearsona. Niech argumentem wejściowym będzie poziom ufności, liczba sukcesów i liczba prób lub poziom ufności i wektor danych (funkcja powinna obsługiwać oba przypadki).

W zadaniu tym zajęłyśmy się wyznaczeniem realizacji przedziału ufności Cloppera-Pearsona. W tym celu wykorzystałyśmy poniższy algorytm, który jako argumenty wejściowe przyjmuje poziom ufności, liczbę sukcesów i liczbę prób:

1. Sprawdzamy czy została podana liczba prób. Jeśli nie to za dane wejściowe przyjmujemy wektor, w którym sukcesy są oznaczone jako "TAK".

   (a) Liczymy liczbę sukcesów, a następnie liczbę prób.

2. Obliczamy przedziały ufności Cloppera-Pearsona korzystając z rozkładu beta z parametrami zależnymi od liczby sukcesów i prób.

3. Zwracamy otrzymane przedziały.

```
intervals_clopper_pearson <- function(alpha, successes, trials = NULL) {

  if (is.null(trials)) {
    data <- successes
```

```
    successes <- sum(data == "TAK")
    trials <- length(data)
  }

  result_lower <- qbeta(alpha/2, successes, trials - successes + 1)
  result_upper <- qbeta(1 - (alpha/2), successes + 1, trials - successes)

  result <- data.frame(
    lower = result_lower,
    upper = result_upper
  )

  return(result)
}
```

Zaproponowaną przez nas funkcję wykorzystamy w dalszej części raportu.

5.2 Zadanie 7

Korzystając z funkcji napisanej w zadaniu 6 (5.1). wyznacz realizacje przedziałów ufności dla prawdopodobieństwa, że pracownik jest zadowolony z wynagrodzenia w pierwszym badanym okresie oraz w drugim badanym okresie. Skorzystaj ze zmiennych **CZY_ZADW** oraz **CZY_ZADW_2** (utwórz zmienną analogicznie jak w zadaniu 3.8). Przyjmij $1-\alpha = 0.95$.

```
df$CZY_ZADOW_2 <- ifelse(df$PYT_3 %in% c("zdecydowanie się nie zgadzam",
    "nie zgadzam się"), "NIE",
                         "TAK")

alph <- 0.05

clopper_pearson_PYT_3 <- intervals_clopper_pearson(alph, df$CZY_ZADOW_2)
clopper_pearson_PYT_2 <- intervals_clopper_pearson(alph, df$CZY_ZADOW)

print(clopper_pearson_PYT_3)
print(clopper_pearson_PYT_2)
```

Wyniki działania funkcji z zadania 5.1 dla zmiennych **PYT_2** i **PYT_3** zostały przedstawione w poniższej tabeli.

Zmienna	Dolna granica przedziału ufności	Górna granica przedziału ufności
PYT_2	0.4583305	0.6007671
PYT_3	0.5184216	0.6588694

Tabela 19: Przedziały ufności dla zmiennej PYT_2 i PYT_3

5.3 Zadanie 8

Zapoznaj się z funkcjami *rbinom* z biblioteki *stats* oraz *binom.confint* z biblioteki *binom*.

```
help(rbinom)
help(binom.confint)
```

W zadaniu 9 (5.4) będziemy używać funkcji *rbinom* z biblioteki *stats* oraz *binom.confint* z biblioteki *binom*.

Pierwsza z nich służy do generowania losowych próbek z rozkładu dwumianowego opisującego liczbę sukcesów w sekwencji niezależnych prób Bernoulliego. Jako wynik zwraca wektor losowych liczb całkowitych, które reprezentują liczbę sukcesów w każdej próbie. Składnia funkcji *rbinom* ma następującą postać: $rbinom(n, size, prob)$, gdzie:

- n - rozmiar próby (np. ilość pytań w ankiecie),
- $size$ - ilość powtórzeń (np. ilość ankietowanych)
- $prom$ - prawdopodobieństwo sukcesu w każdej z prób.

Natomiast funkcja *binom.confint* z biblioteki *binom* służy do obliczania przedziałów ufności dla parametrów rozkładu dwumianowego. Jako wynik zwraca przedział lub przedziały ufności dla prawdopodobieństwa sukcesu w rozkładzie dwumianowym. Składnia funkcji *rbinom* ma następującą postać: $binom.confint(x, n, conf.level, methods)$, gdzie:

- x - liczba sukcesów w próbie,
- n - rozmiar próby,
- $conf.level$ - poziom ufności dla przedziałów ufności,
- $methods$ - metoda lub metody obliczania przedziałów ufności.

Metody obliczania przedziałów ufności dostępne w funkcji *binom.confint* to między innymi: metoda Clopper-Pearsona, metoda Wilsona czy metoda Wilsona-Score'a.

5.4 Zadanie 9

Przeprowadź symulacje, których celem jest porównanie prawdopodobieństwa pokrycia i długości przedziałów ufności Cloppera-Pearsona, Walda i trzeciego dowolnego typu zaimplementowanego w funkcji *binom.confint*. Rozważ $1 - \alpha = 0.95$, rozmiar próby $n \in \{30, 100, 1000\}$ i różne wartości prawdopodobieństwa p. Wyniki umieść na wykresach i sformułuj wnioski, które dla konkretnych danych ułatwią wybór konkretnego typu przedziału ufności.

W celu przeprowadzenia symulacji, których celem jest porównanie prawdopodobieństwa pokrycia i długości przedziałów ufności Cloppera-Pearsona, Walda i Wilsona wykorzystałyśmy poniższy algorytm:

1. Inicjalizacja zmiennych:
   (a) Ustalamy poziom istotności α.
   (b) Określamy liczbę prób n.
   (c) Definiujemy wektor p zawierający prawdopodobieństwa sukcesu, dla których będziemy liczyć przedziały ufności.
   (d) Określamy liczbę symulacji M.
2. Właściwa symulacja:

(a) Dla każdego $n \in 30, 100, 1000$:

i. Dla każdej wartości prawdopodobieństwa sukcesu p:

A. Dla każdej z M symulacji:

- Generujemy próbę z rozkładu dwumianowego.
- Obliczamy przedziały ufności Cloppera-Pearsona, Walda i Wilsona dla liczby sukcesów w próbce.
- Obliczamy długość każdego przedziału ufności, jako różnicę między górną i dolną granicą przedziału.
- Sprawdzamy, czy prawdopodobieństwo sukcesu znajduje się w wyznaczonym przedziale ufności.

B. Obliczamy średnią długość przedziałów ufności dla każdej metody.

C. Obliczamy średnie prawdopodobieństwo pokrycia dla każdej metody.

3. Zwracamy średnie długości przedziałów ufności i średnie prawdopodobieństwo pokrycia dla każdej metody.

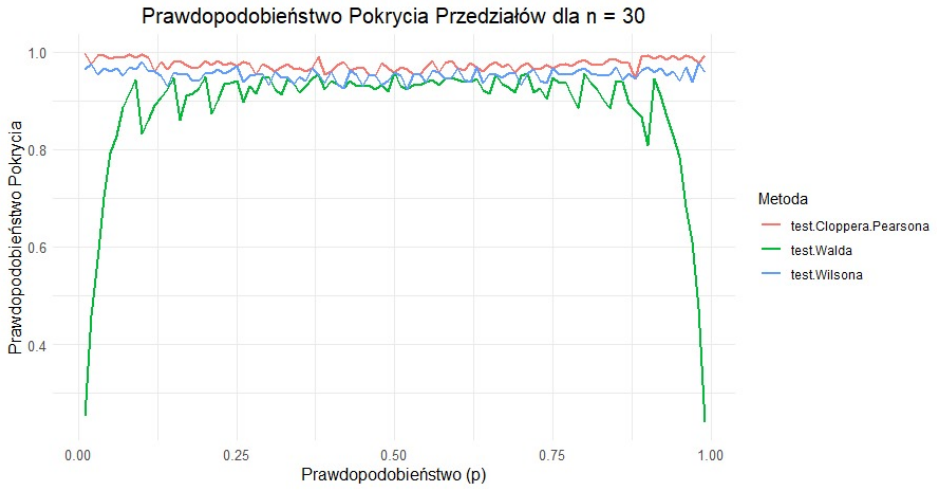
Poniżej widoczny jest kod obrazujący działanie powyższego algorytmu.

```r
calculate_cf <- function(alpha, n, p, M) {
  size <- length(p)

  n_cp <- numeric(size)
  n_wald <- numeric(size)
  n_wilson <- numeric(size)

  p_cp <- numeric(size)
  p_wald <- numeric(size)
  p_wilson <- numeric(size)

    for (k in 1:size) {
      length_cp <- numeric(M)
      length_wald <- numeric(M)
      length_wilson <- numeric(M)

      coverage_cp <- numeric(M)
      coverage_wald <- numeric(M)
      coverage_wilson <- numeric(M)

      for (i in 1:M) {
        sample_binom <- rbinom(n, 1, p[k])
        successes <- sum(sample_binom)

        clopper_pearson <- binom.confint(successes, n, conf.level = 1 - alpha, methods = "exact")
        wald <- binom.confint(successes, n, conf.level = 1 - alpha, methods = "asymptotic")
        wilson <- binom.confint(successes, n, conf.level = 1 - alpha, methods = "wilson")
        length_cp[i] <- clopper_pearson[, "upper"] - clopper_pearson[, "lower"]
        length_wald[i] <- wald[, "upper"] - wald[, "lower"]
        length_wilson[i] <- wilson[, "upper"] - wilson[, "lower"]

        coverage_cp[i] <- (clopper_pearson[,"lower"] <= p[k]) & (p[k] <= clopper_pearson[,"upper"])
```

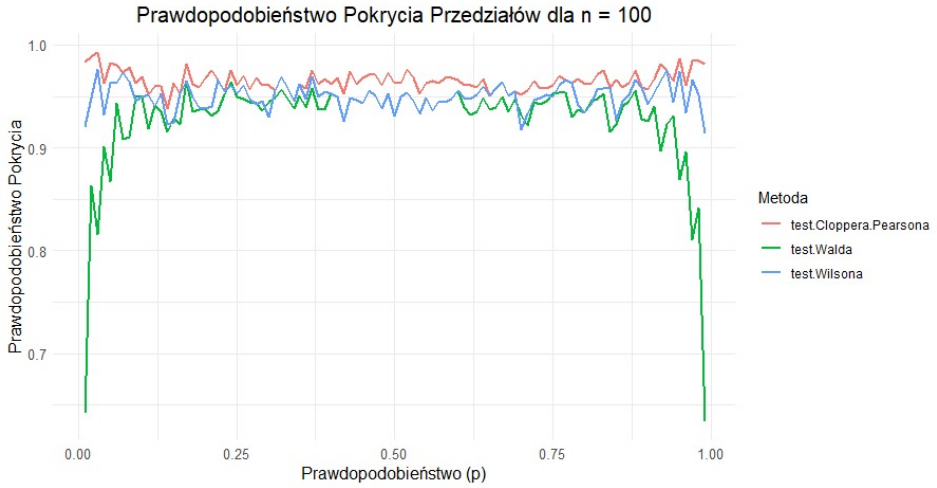
```
        coverage_wald[i] <- (wald[,"lower"] <= p[k]) & (p[k] <= wald[,"upper"])
        coverage_wilson[i] <- (wilson[,"lower"] <= p[k]) & (p[k] <= wilson[,"upper"])
      }

      n_cp[k] <- mean(length_cp)
      n_wald[k] <- mean(length_wald)
      n_wilson[k] <- mean(length_wilson)

      p_cp[k] <- mean(coverage_cp)
      p_wald[k] <- mean(coverage_wald)
      p_wilson[k] <- mean(coverage_wilson)
    }

  result_n <- data.frame(
    "p" = p,
    "test Cloppera-Pearsona" = n_cp,
    "test Walda" = n_wald,
    "test Wilsona" =  n_wilson
  )

  result_p <- data.frame(
    "p" = p,
    "test Cloppera-Pearsona" =  p_cp,
    "test Walda" =  p_wald,
    "test Wilsona" =  p_wilson
  )

  return(list(result_n = result_n, result_p = result_p))
}

alpha <- 0.05
n <- c(30, 100, 1000)
p <- seq(0.01, 0.99, by = 0.01)
M <- 100

results_30 <- calculate_cf(alpha, n[1], p, M)
results_100 <- calculate_cf(alpha, n[2], p, M)
results_1000 <- calculate_cf(alpha, n[3], p, M)

df_length_30 <- melt(results_30$result_n, id.vars = "p", variable.name = "Method", value.name = "Length")
df_coverage_30 <- melt(results_30$result_p, id.vars = "p", variable.name = "Method", value.name = "Coverage")
df_length_100 <- melt(results_100$result_n, id.vars = "p", variable.name = "Method", value.name = "Length")
df_coverage_100 <- melt(results_100$result_p, id.vars = "p", variable.name = "Method", value.name = "Coverage")
df_length_1000 <- melt(results_1000$result_n, id.vars = "p", variable.name = "Method", value.name = "Length")
df_coverage_1000 <- melt(results_1000$result_p, id.vars = "p", variable.name = "Method", value.name = "Coverage")
```

Poniżej widoczny jest kod służący do wizualizacji prawdopodobieństw pokrycia przedziałów ufności Cloppera-Pearsona, Walda i Wilsona dla liczby prób n równej odpowiednio 30, 100 i 1000 i parametru $p = 0.01, 0.02, ..., 0.99$.

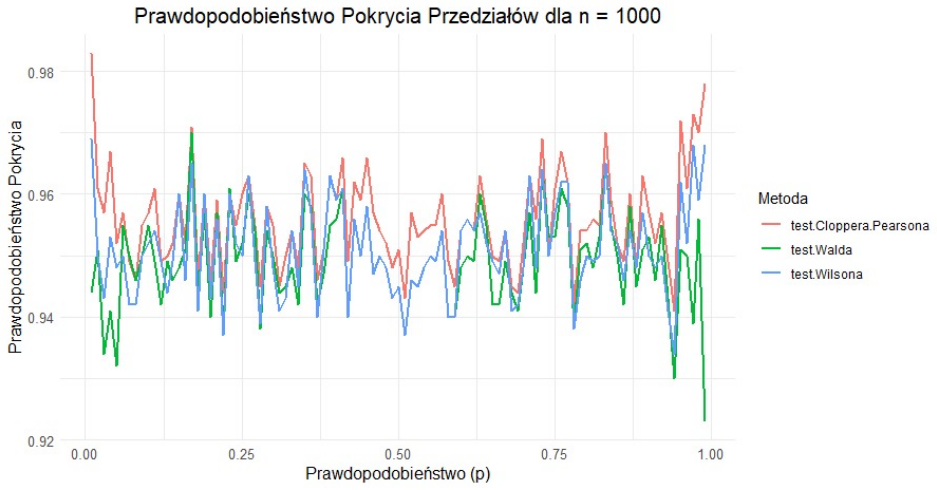
```r
# wykresy prawdopodobieństwa pokrycia (czy)
ggplot(df_coverage_30, aes(x = p, y = Coverage, color = Method)) +
  geom_line(linewidth = 0.8) +
  labs(title = "Prawdopodobieństwo Pokrycia Przedziałów dla n = 30",
       y = "Prawdopodobieństwo Pokrycia",
       x = "Prawdopodobieństwo (p)",
       color = "Metoda") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))


ggplot(df_coverage_100, aes(x = p, y = Coverage, color = Method)) +
  geom_line(linewidth = 0.8) +
  labs(title = "Prawdopodobieństwo Pokrycia Przedziałów dla n = 100",
       y = "Prawdopodobieństwo Pokrycia",
       x = "Prawdopodobieństwo (p)",
       color = "Metoda") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))


ggplot(df_coverage_1000, aes(x = p, y = Coverage, color = Method)) +
  geom_line(linewidth = 0.8) +
  labs(title = "Prawdopodobieństwo Pokrycia Przedziałów dla n = 1000",
       y = "Prawdopodobieństwo Pokrycia",
       x = "Prawdopodobieństwo (p)",
       color = "Metoda") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))
```



Rysunek 16: Prawdopodobieństwo pokrycia przedziałów dla n = 30



Rysunek 17: Prawdopodobieństwo pokrycia przedziałów dla n = 100



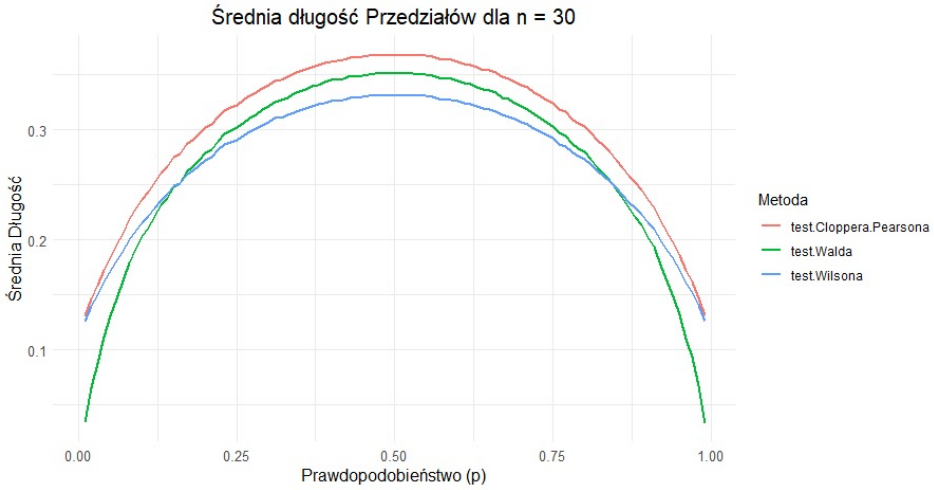
Rysunek 18: Prawdopodobieństwo pokrycia przedziałów dla n = 1000

Na wykresach 16, 17, 18 zwizualizowane zostały prawdopodobieństwa pokrycia przedziałów ufności Cloppera-Pearsona, Walda i Wilsona dla liczby prób n równej odpowiednio 30, 100 i 1000 i parametru $p = 0.01, 0.02, ..., 0.99$.

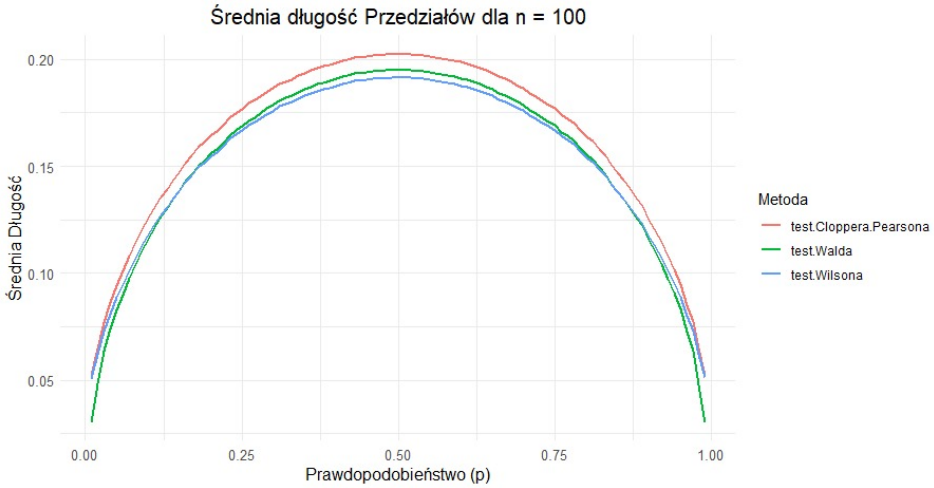
Natomiast poniżej widoczny jest kod służący do wizualizacji średniej długości przedziałów ufności Cloppera-Pearsona, Walda i Wilsona dla liczby prób n równej odpowiednio 30, 100 i 1000 i parametru $p = 0.01, 0.02, ..., 0.99$.

```
# wykresy średniej długości
ggplot(df_length_30, aes(x = p, y = Length, color = Method)) +
  geom_line(linewidth = 0.8) +
  labs(title = "Średnia długość Przedziałów dla n = 30",
       y = "Średnia Długość",
       x = "Prawdopodobieństwo (p)",
       color = "Metoda") +
  theme_minimal()+
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))

ggplot(df_length_100, aes(x = p, y = Length, color = Method)) +
  geom_line(linewidth = 0.8) +
  labs(title = "Średnia długość Przedziałów dla n = 100",
       y = "Średnia Długość",
       x = "Prawdopodobieństwo (p)",
       color = "Metoda") +
  theme_minimal()+
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))

ggplot(df_length_1000, aes(x = p, y = Length, color = Method)) +
  geom_line(linewidth = 0.8) +
  labs(title = "Średnia długość Przedziałów dla n = 1000",
       y = "Średnia Długość",
       x = "Prawdopodobieństwo (p)",
```

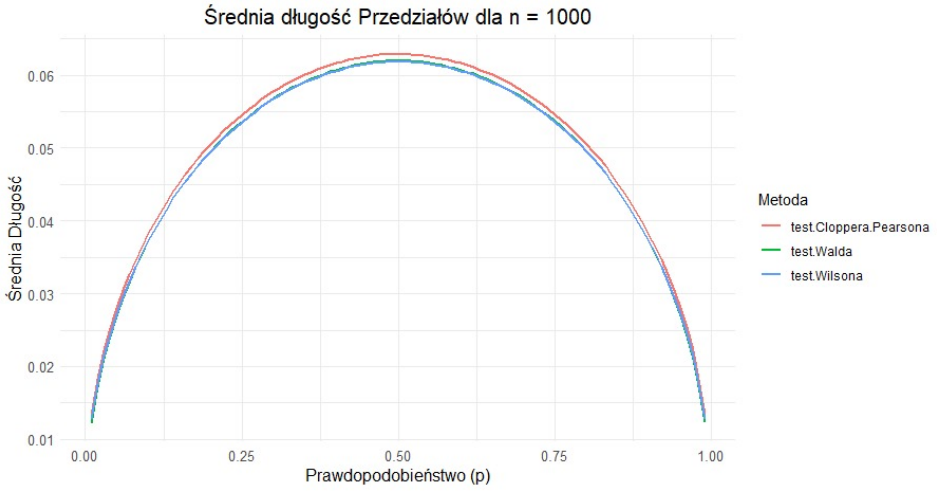
```
      color = "Metoda") +
theme_minimal()+
theme(
  axis.title = element_text(size = 12),
  axis.text = element_text(size = 10),
  plot.title = element_text(size = 15, hjust = 0.5))
```



Rysunek 19: Średnia długość przedziałów przedziałów dla n = 30



Rysunek 20: Średnia długość przedziałów przedziałów dla n = 100



Rysunek 21: Średnia długość przedziałów przedziałów dla n = 1000

Na wykresach 19, 20, 21 zwizualizowane zostały średnie długości przedziałów ufności Cloppera-Pearsona, Walda i Wilsona dla liczby prób n równej odpowiednio 30, 100 i 1000 i parametru $p = 0.01, 0.02, ..., 0.99$.

Na podstawie analizy wykresów 16 - 21, można stwierdzić, że metoda konstruowania przedziałów ufności Cloppera-Pearsona wykazuje największą dokładność dla środkowych wartości prawdopodobieństwa p oraz dla małych prób. W porównaniu do innych metod, ta metoda często osiąga wyższe prawdopodobieństwo pokrycia, co świadczy o uzyskaniu wysokiej dokładności w konstruowaniu przedziałów ufności, szczególnie dla małych prób.

Metoda Wilsona również okazała się skuteczna, prawdopodobieństwo pokrycia przedziału ufności jest dla niej również dość wysokie. Metoda ta nie jest tak konserwatywna jak metoda Cloppera-Pearsona, częściej niedoszacowuje wartości.

Z kolei metoda Walda okazała się najmniej skuteczna dla małych prób i skrajnych wartości prawdopodobieństwa, ale dla większych prób osiąga podobną skuteczność jak pozostałe metody.

Analizując średnie długości przedziałów ufności wyznaczone za pomocą metod Cloppera-Pearsona, Walda i Wilsona, dla różnych wielkości prób n i wartości prawdopodobieństw p, można zauważyć, że średnie długości przedziałów są najmniejsze dla skrajnych wartości prawdopodobieństwa oraz maleją wraz ze wzrostem wielkości próby (n). Największa różnica pomiędzy średnimi długościami przedziałów występuje dla małych prób ($n = 30$), natomiast dla dużych prób ($n = 1000$) różnice te są mniejsze.

Długości przedziałów różnią się między metodami, ale są do siebie zazwyczaj dość zbliżone. Wartości prawdopodobieństw również są do siebie zbliżone. Oznacza to, że różne metody symulacji przedziałów ufności prowadzą do podobnych wyników średnich (z przewagą w stronę metody Cloppera-Pearsona).

6 Część V

6.1 Zadanie 10

Zapoznaj się z funkcjami *binom.test* oraz *prop.test* z biblioteki *stats*.

```
help(binom.test) # test dokładny
help(prop.test) # test asymptotyczny
# correcter = TRUE - poprawione
```

W zadaniu 11 (6.2) oraz 12 (6.3) będziemy używać funkcji *binom.test* oraz *prop.test* z biblioteki *stats*.

Pierwsza z nich służy do przeprowadzania testu hipotez dotyczących rozkładu dwumianowego. Jej składnia ma następującą postać: $binom.test(x, n, p, alternative, conf.level)$, gdzie:

- x - liczba sukcesów w próbie,
- n - rozmiar próby,
- p - spodziewane prawdopodobieństwo sukcesu (wartość opcjonalna, jeśli nie jest podana to zostaje użyta wartość próbkowa (x/n),
- $alternative$ - hipoteza alternatywna przyjmująca możliwe wartości: "two.sided dwustronny, "less jedenstronny mniejszy i "greater jedenstronny większy,
- $conf.level$ - poziom ufności dla przedziału ufności (domyślnie 0.95).

W wyniku działania tej funkcji otrzymujemy wyniki testu wraz z wartościami p i przedziałem ufności.

Natomiast funkcja *prop.test* służy do przeprowadzania testu hipotez dotyczących różnicy proporcji w dwóch grupach. Jej składnia ma następującą postać:

$$prop.test(x, n, correct, alternative, conf.level),$$

gdzie:

- x - wektor zawierający liczby sukcesów w próbie,
- n - wektor zawierający rozmiary prób,
- $correct$ - określa czy ma być stosowana poprawka na ciągłość zalecana dla małych prób,
- $alternative$ - hipoteza alternatywna przyjmująca możliwe wartości: "two.sided " dwustronny, "less" jedenstronny mniejszy i "greater" jedenstronny większy,
- $conf.level$ - poziom ufności dla przedziału ufności (domyślnie 0.95).

6.2 Zadanie 11

Dla danych z pliku "ankieta.csv"korzystając z funkcji z zadania 10 (6.1), przyjmując $1-\alpha = 0.95$, zweryfikuj następujące hipotezy i sformułuj wnioski:

1. Prawdopodobieństwo, że w firmie pracuje kobieta wynosi 0.5.
2. Prawdopodobieństwo, że pracownik jest zadowolony ze swojego wynagrodzenia w pierwszym badanym okresie jest większe bądź równe 0.7.
3. Prawdopodobieństwo, że kobieta pracuje na stanowisku menedżerskim jest równe prawdopodobieństwu, że mężczyzna pracuje na stanowisku menedżerskim.
4. Prawdopodobieństwo, że kobieta jest zadowolona ze swojego wynagrodzenia w pierwszym badanym okresie jest równe prawdopodobieństwu, że mężczyzna jest zadowolony ze swojego wynagrodzenia w pierwszym badanym okresie.
5. Prawdopodobieństwo, że kobieta pracuje w dziale obsługi kadrowo-płacowej jest większe lub równe prawdopodobieństwu, że mężczyzna pracuje w dziale obsługi kadrowo-płacowej.

Dla każdej z hipotez przeprowadziliśmy *binom.test*, *prop.test* bez poprawki oraz *prop.test* z poprawką na ciągłość. Następnie sprawdziłyśmy czy otrzymane p-wartości są większe od poziomu istotności α, który w naszym przypadku wynosi 0.05.

Przypominając: jeżeli poziom krytyczny (p-wartość) jest mniejszy bądź równy od poziomu istotności α to hipotezę odrzucamy, w przeciwnym przypadku nie mamy podstaw do jej odrzucenia.

```r
# dla części z podpunktów
females <- sum(df$PŁEĆ == "K")
males <- sum(df$PŁEĆ == "M")
length_sex <- c(females, males)
all_rows <- nrow(df)
```

- HIPOTEZA 1: Prawdopodobieństwo, że w firmie pracuje kobieta wynosi 0.5.

```r
# 1.Prawdopodobieństwo, że w firmie pracuje kobieta wynosi 0.5.
binom_test <- binom.test(females, all_rows, p=0.5, alternative = "two.
    sided", conf.level = 0.95)
prop_test_1 <- prop.test(females, all_rows, p=0.5, alternative = "two.
    sided", conf.level = 0.95, correct = TRUE)
prop_test_2 <- prop.test(females, all_rows, p=0.5, alternative = "two.
    sided", conf.level = 0.95, correct = FALSE)

print(binom_test$p.value)
print(prop_test_1$p.value)
print(prop_test_2$p.value)
```

W przypadku tej hipotezy otrzymaliśmy następujące p-wartości:

- *binom.test*: $4.972973 * 10^{-5}$
- *prop.test* bez poprawki: $5.565628 * 10^{-5}$
- *prop.test* z poprawką: $4.109788 * 10^{-5}$

Dla każdego z testów p-wartość jest mniejsza niż poziom istotności $\alpha = 0.05$ co sugeruje, że należy odrzucić hipotezę zerową. Sugeruje to, że prawdopodobieństwo, że w firmie pracuje kobieta jest istotnie statystycznie różne od 0.5.

- HIPOTEZA 2: Prawdopodobieństwo, że pracownik jest zadowolony ze swojego wynagrodzenia w pierwszym badanym okresie jest większe bądź równe 0.7.

```r
# 2.Prawdopodbieństwo, że pracownik jest zadowolony ze swojego
    wynagrodzenia w pierwszym badanym okresie jest większe bądź równe 0.7.
content_1 <- sum(df$CZY_ZADOW == "YES")

binom_test <- binom.test(content_1, all_rows, p=0.7, alternative = "
    greater", conf.level = 0.95)
prop_test_1 <- prop.test(content_1, all_rows, p=0.7, alternative = "
    greater", conf.level = 0.95, correct = TRUE)
prop_test_2 <- prop.test(content_1, all_rows, p=0.7, alternative = "
    greater", conf.level = 0.95, correct = FALSE)

print(binom_test$p.value)
print(prop_test_1$p.value)
print(prop_test_2$p.value)

content_2 <- sum(df$CZY_ZADOW_2 == "YES")

binom_test <- binom.test(content_2, all_rows, p=0.7, alternative = "
    greater", conf.level = 0.95)
prop_test_1 <- prop.test(content_2, all_rows, p=0.7, alternative = "
    greater", conf.level = 0.95, correct = TRUE)
prop_test_2 <- prop.test(content_2, all_rows, p=0.7, alternative = "
    greater", conf.level = 0.95, correct = FALSE)

print(binom_test$p.value)
print(prop_test_1$p.value)
print(prop_test_2$p.value)
```

W przypadku tej hipotezy, zarówno dla zmiennej **CZY_ZADOW**, jak i dla zmiennej **CZY_ZADOW_2**, dla wszystkich testów otrzymaliśmy p-wartość równą 1. Pozwala nam to na przyjęcie hipotezy zerowej.

- HIPOTEZA 3: Prawdopodobieństwo, że kobieta pracuje na stanowisku menedżerskim jest równe prawdopodobieństwu, że mężczyzna pracuje na stanowisku menedżerskim.

```r
# 3. Prawdopodobieństwo, że kobieta pracuje na stanowisku menedżerskim
    jest równe prawdopodobieństwu, że mężczyzna pracuje na stanowisku
    menedżerskim.
manager <- filter(df, CZY_KIER == "Tak")
female_manager <- sum(manager$PŁEĆ == "K")
male_manager <- sum(manager$PŁEĆ == "M")

fem_male_manager <- c(female_manager, male_manager)

prop_test_1 <- prop.test(fem_male_manager, length_sex, alternative = "two
    .sided", conf.level = 0.95, correct = TRUE)
prop_test_2 <- prop.test(fem_male_manager, length_sex, alternative = "two
    .sided", conf.level = 0.95, correct = FALSE)

print(prop_test_1$p.value)
print(prop_test_2$p.value)
```

W przypadku tej hipotezy otrzymaliśmy następujące p-wartości:

- *prop.test* bez poprawki: 0.6389361

- *prop.test* z poprawką: 0.4930904

Możemy zauważyć, że p-wartości dla obu testów są większe od poziomu istotności $\alpha = 0.05$. Pozwala nam to przyjąć hipotezę zerową i stwierdzić, że najprawdopodobniej nie istnieje istotna statystycznie różnica pomiędzy zajmowaniem stanowisk menadżerskich przez kobiety i mężczyzn.

- HIPOTEZA 4: Prawdopodobieństwo, że kobieta jest zadowolona ze swojego wynagrodzenia w pierwszym badanym okresie jest równe prawdopodobieństwu, że mężczyzna jest zadowolony ze swojego wynagrodzenia w pierwszym badanym okresie.

```
# 4. Prawdopodobieństwo, że kobieta jest zadowolona ze swojego
   wynagrodzenia w pierwszym badanym okresie jest równe prawdopodobień
   stwu, że mężczyzna jest zadowolony ze swojego wynagrodzenia w
   pierwszym badanym okresie.
content <- filter(df, CZY_ZADOW == "TAK")
female_content <- sum(content$PŁEĆ == "K")
male_content <- sum(content$PŁEĆ == "M")

fem_male_content <- c(female_content, male_content)

prop_test_1 <- prop.test(fem_male_content, length_sex, alternative = "two
   .sided", conf.level = 0.95, correct = TRUE)
prop_test_2 <- prop.test(fem_male_content, length_sex, alternative = "two
   .sided", conf.level = 0.95, correct = FALSE)

print(prop_test_1$p.value)
print(prop_test_2$p.value)
```

W przypadku tej hipotezy otrzymaliśmy następujące p-wartości:

- *prop.test* bez poprawki: 0.7379521
- *prop.test* z poprawką: 0.6293763

Możemy zauważyć, że p-wartości dla obu testów są większe od poziomu istotności $\alpha = 0.05$. Pozwala nam to przyjąć hipotezę zerową i stwierdzić, że najprawdopodobniej nie istnieje istotna statystycznie różnica pomiędzy zadowoleniem ze swojego wynagrodzenia kobiet oraz mężczyzn w pierwszym badanym okresie.

- HIPOTEZA 5: Prawdopodobieństwo, że kobieta pracuje w dziale obsługi kadrowo-płacowej jest większe lub równe prawdopodobieństwu, że mężczyzna pracuje w dziale obsługi kadrowo-płacowej.

```
# 5. Prawdopodobieństwo, że kobieta pracuje w dziale obsługi kadrowo-pł
   acowej jest większe lub równe prawdopodobieństwu, że mężczyzna pracuje
    w dziale obsługi kadrowo-płacowej.
HR <- filter(df, DZIAŁ == "HR")
female_HR <- sum(HR$PŁEĆ == "K")
male_HR <- sum(HR$PŁEĆ == "M")

fem_male_HR <- c(female_HR, male_HR)

prop_test_1 <- prop.test(fem_male_HR, length_sex, alternative = "greater"
   , conf.level = 0.95, correct = TRUE)
prop_test_2 <- prop.test(fem_male_HR, length_sex, alternative = "greater"
   , conf.level = 0.95, correct = FALSE)
print(prop_test_1$p.value)
print(prop_test_2$p.value)
```

W przypadku tej hipotezy otrzymaliśmy następujące p-wartości:

- *prop.test* bez poprawki: 0.9960475
- *prop.test* z poprawką: 0.9978835

Możemy zauważyć, że p-wartości dla obu testów są większe od poziomu istotności $\alpha = 0.05$ i bliskie 1. Pozwala nam to przyjąć hipotezę zerową i stwierdzić, że prawdopodobieństwo, że kobieta pracuje w dziale obsługi kadrowo-płacowej jest większe lub równe prawdopodobieństwu, że mężczyzna pracuje w tym dziale.

6.3 Zadanie 12

Wyznacz symulacyjnie moc testu dokładnego oraz moc testu asymptotycznego w przypadku weryfikacji hipotezy zerowej $H_0 : p = 0.9$ przeciwko $H_1 : p \neq 0.9$ przyjmując wartość $1 - \alpha = 0.95$. Uwzględnij różne wartości alternatyw i różne rozmiary próby. Sformułuj wnioski.

W tym zadaniu zajęłyśmy się wyznaczeniem mocy testu dokładnego oraz asymptotycznego z poprawką i bez dla $n \in \{30, 100, 1000\}$ i $p = 0.01, 0.02, ..., 0.89, 0.91, ..., 0.99$. Dla przypomnienia: Moc testu to prawdopodobieństwo odrzucenia fałszywej hipotezy zerowej i przyjęcia prawdziwej hipotezy alternatywnej.

W celu obliczenia mocy testów wykorzystałyśmy poniższy algorytm:

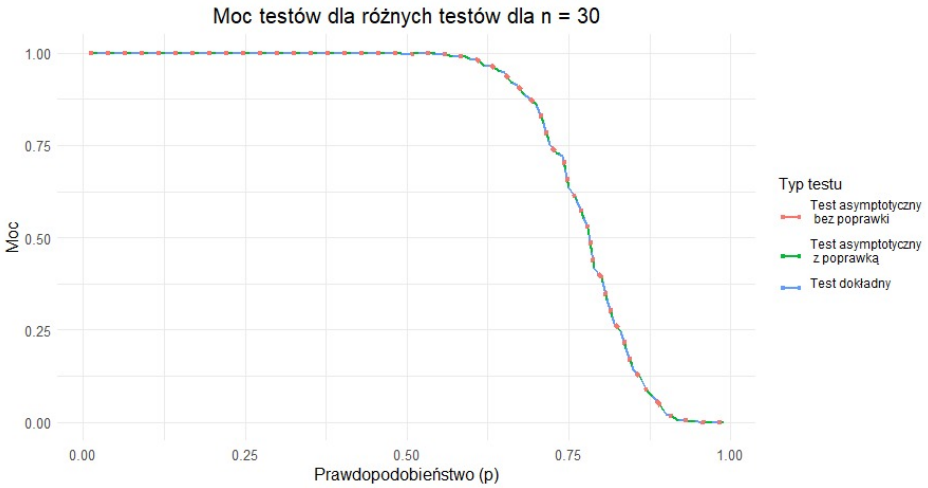
1. Inicjalizacja zmiennych:
   (a) Określenie wartości hipotezy zerowej H_0.
   (b) Ustalenie poziomu istotności α.
   (c) Określenie wielkości próby n.
   (d) Zdefiniowanie wektora prawdopodobieństw p.
   (e) Określenie liczby symulacji M.
2. Obliczenie mocy testu:
   (a) Dla każdego n:
      i. Dla każdej wartości prawdopodobieństwa sukcesu p:
         A. Dla każdej z M symulacji:
            - Generujemy próbkę o rozmiarze n z rozkładu dwumianowego, w tym celu używamy funkcji *rbinom*.
            - Przeprowadzamy testy z użyciem funkcji *binom.test* i *prop.test*.
            - Zapisujemy czy odrzuciliśmy hipotezę zerową czy nie.
         B. Obliczamy średnią z liczby odrzuconych hipotez zerowych dla każdego z testów.
      ii. Zapisujemy wyniki
   (b) Przedstawiamy wyniki za pomocą wykresów.

```r
calculate_power <- function(H_0, alpha, n, p, M) {
  size <- length(p)

  result_binom <- numeric(size)
  result_prop_1 <- numeric(size)
  result_prop_2 <- numeric(size)

  for (k in 1:size) {
    rejection_binom <- 0
    rejection_prop_1 <- 0
    rejection_prop_2 <- 0

    for (i in 1:M) {
      sample_binom <- rbinom(n, 1, p[k])
      summ <- sum(sample_binom)

      binom <- binom.test(summ, n, p=H_0, alternative = "two.sided", conf.level = 1 - alpha)
      prop_1 <- prop.test(summ, n, p=H_0, alternative = "two.sided", conf.level = 1 - alpha, correct = TRUE)
      prop_2 <- prop.test(summ, n, p=H_0, alternative = "two.sided", conf.level = 1 - alpha, correct = FALSE)

      binom_pv <- binom$p.value
      prop_pv_1 <- prop_1$p.value
      prop_pv_2 <- prop_2$p.value

      if (binom_pv < alpha) {
        rejection_binom <- rejection_binom + 1
      }
      if (prop_pv_1 < alpha) {
        rejection_prop_1 <- rejection_prop_1 + 1
      }
      if (prop_pv_2 < alpha) {
        rejection_prop_2 <- rejection_prop_2 + 1
      }

    }
    result_binom[k] <- rejection_binom/M
    result_prop_1[k] <- rejection_prop_1/M
    result_prop_2[k] <- rejection_prop_2/M
  }
  return(list(result_binom = result_binom, result_prop_1 = result_prop_1, result_prop_2 = result_prop_2))
}

alpha <- 0.05
H_0 <- 0.9
n <- c(30, 100, 1000)
p <- seq(0.01, 0.99, by = 0.01)
M <- 1000

result_30 <- calculate_power(H_0, alpha, n[1], p, M)
result_100 <- calculate_power(H_0, alpha, n[2], p, M)
result_1000 <- calculate_power(H_0, alpha, n[3], p, M)

df_30 <- data.frame(p = p, moc_binom = result_30$result_binom, moc_prop_1 = result_30$result_prop_1, moc_prop_2 = result_30$result_prop_2)
df_100 <- data.frame(p = p, moc_binom = result_100$result_binom, moc_prop
```

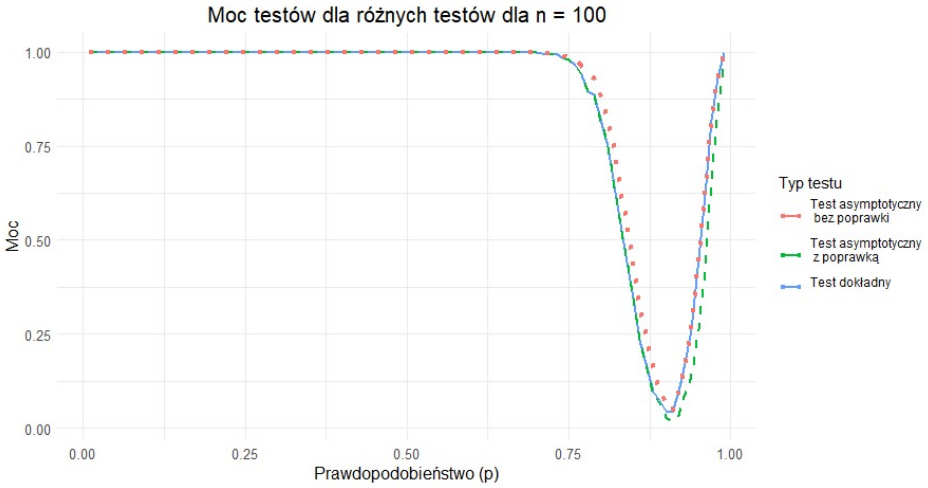
```
    _1 = result_100$result_prop_1, moc_prop_2 = result_100$result_prop_2)
df_1000 <- data.frame(p = p, moc_binom = result_1000$result_binom, moc_
    prop_1 = result_1000$result_prop_1, moc_prop_2 = result_1000$result_
    prop_2)
```

Poniżej widoczne są kody służące do wizualizacji wyników odpowiednio dla $n = 30$, $n = 100$ i $n = 1000$.

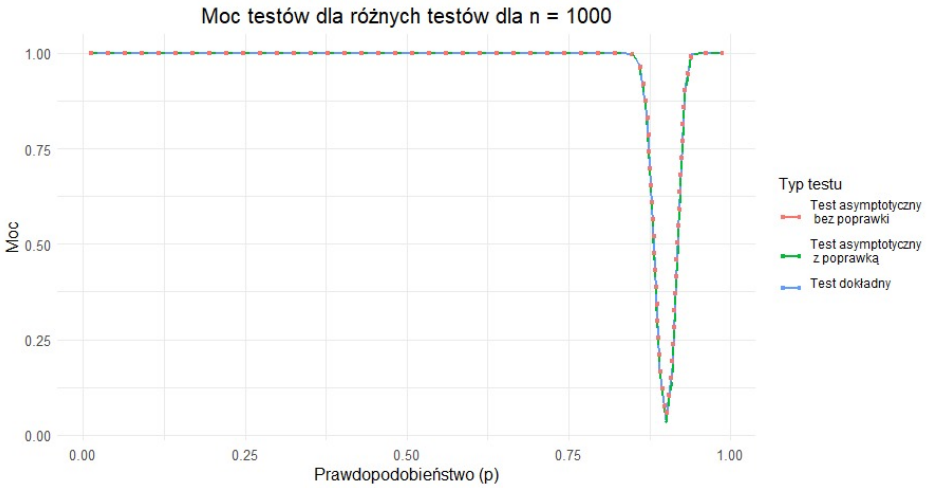
```
ggplot(df_30, aes(x = p)) +
  geom_line(aes(y = moc_binom, color = "Test dokładny\n"), linewidth =
    0.8) +
  geom_line(aes(y = moc_prop_1, color = "Test asymptotyczny\n z poprawką\
    n"), linetype = "dashed", linewidth = 1) +
  geom_line(aes(y = moc_prop_2, color = "Test asymptotyczny\n z bez
    poprawki\n"), linetype = "dotted", linewidth = 1.5) +
  labs(title = "Moc testów dla różnych testów dla n = 30",
       x = "Prawdopodobieństwo (p)",
       y = "Moc",
       color = "Typ testu") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))

ggplot(df_100, aes(x = p)) +
  geom_line(aes(y = moc_binom, color = "Test dokładny\n"), linewidth =
    0.8) +
  geom_line(aes(y = moc_prop_1, color = "Test asymptotyczny\n z poprawką\
    n"), linetype = "dashed", linewidth = 1) +
  geom_line(aes(y = moc_prop_2, color = "Test asymptotyczny\n z bez
    poprawki\n"), linetype = "dotted", linewidth = 1.5) +
  labs(title = "Moc testów dla różnych testów dla n = 100",
       x = "Prawdopodobieństwo (p)",
       y = "Moc",
       color = "Typ testu") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))

ggplot(df_1000, aes(x = p)) +
  geom_line(aes(y = moc_binom, color = "Test dokładny\n"), linewidth =
    0.8) +
  geom_line(aes(y = moc_prop_1, color = "Test asymptotyczny\n z poprawką\
    n"), linetype = "dashed", linewidth = 1) +
  geom_line(aes(y = moc_prop_2, color = "Test asymptotyczny\n z bez
    poprawki\n"), linetype = "dotted", linewidth = 1.5) +
  labs(title = "Moc testów dla różnych testów dla n = 1000",
       x = "Prawdopodobieństwo (p)",
       y = "Moc",
       color = "Typ testu") +
  theme_minimal() +
  theme(
    axis.title = element_text(size = 12),
    axis.text = element_text(size = 10),
    plot.title = element_text(size = 15, hjust = 0.5))
```



Rysunek 22: Moc testu dla różnych testów dla n = 30



Rysunek 23: Moc testu dla różnych testów dla n = 100



Rysunek 24: Moc testu dla różnych testów dla n = 1000

Analizując wygląd wykresów 22, 23, 24 możemy zauważyć, że wraz ze wzrostem n dołek dokładniej pokrywa się z wartością prawdopodobieństwa z hipotezy zerowej, równą 0.9.

Podsumowując, na podstawie analizy wyników możemy wnioskować, że nie istnieje test najlepszy w każdej sytuacji. Wybór pomiędzy testem dokładnym i testem asymptotycznym zależy od konkretnych warunków badawczych. Test dokładny będzie lepszym wyborem, jeśli zależy nam na kontrolowaniu ryzyka popełnienia błędu I-go rodzaju. Natomiast w przypadku większych prób oraz gdy zależy nam na większej mocy powinniśmy wybrać testy asymptotyczne.

7 Zadanie dodatkowe

7.1 Zadanie *1

Wyznacz granice asymptotycznego przedziału ufności dla prawdopodobieństwa sukcesu bazując na przekształceniu logit korzystając z metody delta. Zaimplementuj metodę oraz porównaj wyniki z funkcją *binom.confint*.

W tym zadaniu użyłyśmy przekształcenia logitowego, metody delta oraz sposobu, w jaki są one wykorzystywane do wyznaczania asymptotycznych przedziałów ufności dla prawdopodobieństwa sukcesu.

Poniżej znajduje się wyjaśnienie kluczowych pojęć:

- **Przekształcenie logitowe** - przekształcenie logitowe prawdopodobieństwa sukcesu p jest zdefiniowane jako:
$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right),$$
gdzie p - prawdopodobieństwem sukcesu.

- **Metoda delta** - jest to technika wykorzystywana do aproksymacji rozkładu funkcji estymatora. Jeśli mamy estymator $\hat{\theta}$ z wariancją $\text{Var}(\hat{\theta})$ i funkcję g, której chcemy

użyć na $\hat{\theta}$, to wariancja $g(\hat{\theta})$ może być aproksymowana przez:

$$\text{Var}(g(\hat{\theta})) \approx [g'(\hat{\theta})]^2 \text{Var}(\hat{\theta}),$$

gdzie $g'(\hat{\theta})$ - pierwsza pochodna g względem $\hat{\theta}$.

- **Asymptotyczny przedział ufności dla p** - załóżmy, że mamy n prób binarnych (np. sukces/porażka), z których X to liczba sukcesów. Estymatorem p jest $\hat{p} = \frac{X}{n} = \frac{1}{n}\sum_{i=1}^{n} X_i$, $X_i \sim \text{B}(n, p)$. Przekształcenie logitowe tego estymatora to $\text{logit}(\hat{p})$. Aby zastosować metodę delta do wyznaczenia przedziału ufności dla p, potrzebujemy najpierw pochodnej przekształcenia logitowego względem p, co daje:

$$\frac{d}{dp}\text{logit}(p) = \frac{1}{p(1-p)}$$

Zatem wariancja $\text{logit}(\hat{p})$ może być aproksymowana przez:

$$\text{Var}(\text{logit}(\hat{p})) \approx \left[\frac{1}{\hat{p}(1-\hat{p})}\right]^2 \text{Var}(\hat{p}) = \left[\frac{1}{\hat{p}(1-\hat{p})}\right]^2 \frac{\hat{p}(1-\hat{p})}{n} = \frac{1}{n\hat{p}(1-\hat{p})}$$

$(*) Var(\hat{p}) = Var(\frac{1}{n}\sum_{i=1}^{n} X_i) = \frac{1}{n^2}\sum_{i=1}^{n} Var(X_i) \overset{iid}{=} \frac{1}{n^2} n\hat{p}(1-\hat{p}) = \frac{\hat{p}(1-\hat{p})}{n}$
Wtedy:

$$Q = \frac{g(\hat{\theta}) - g(\theta)}{g'(\theta)\sqrt{\frac{Var(\hat{\theta})}{n}}},$$

gdzie: $Q \sim \text{N}(0, 1)$

Mając to, możemy wyznaczyć asymptotyczny przedział ufności dla $\text{logit}(p)$ używając kwantyli rozkładu normalnego $q_{1-\frac{\alpha}{2}}$, gdzie α to poziom istotności.

$$P(\theta \in [-q_{1-\frac{\alpha}{2}}; q_{1-\frac{\alpha}{2}}]) = 1 - \alpha$$

Dokonując odpowiednich przekształceń otrzymujemy:

$$\text{logit}(\hat{p}) \pm q_{1-\alpha/2}\sqrt{\frac{1}{n\hat{p}(1-\hat{p})}}$$

Po rozwinięciu $logit(\hat{p})$ do powyższego wzoru otrzymujemy:

$$logit(p) \in \left[\ln\left(\frac{\hat{p}}{1-\hat{p}}\right) \pm q_{1-\alpha/2}\frac{1}{\sqrt{n\hat{p}(1-\hat{p})}}\right]$$

Aby otrzymać przedział ufności dla p, odwracamy przekształcenie logitowe i podstawiamy do wzoru powyżej:

$$p = \frac{e^{\text{logit}(p)}}{1 + e^{\text{logit}(p)}}$$

Następnie, w celu sprawdzenia poprawności zaimplementowanej przez nas metody dokonałyśmy porównania z funkcją *binom.confint*.

Funkcja ta oferuje kilka metod wyznaczania przedziałów ufności dla p, w tym metodę asymptotyczną, która jest zbliżona do opisanej powyżej, ale może korzystać z różnych aproksymacji. Zaimplementowana przez nas metoda oparta na przekształceniu logitowym i metodzie delta jest jedną z metod asymptotycznych w związku z czym przypuszczamy, że powinna dawać podobne wyniki do tych uzyskanych za pomocą *binom.confint* z odpowiednią opcją metody.

Poniżej widoczny jest kod prezentujący metody opisane powyżej.

```
logit_confint_delta <- function(successes, trials, alpha) {
  p_hat <- successes / trials
  logit_p_hat <- log(p_hat / (1 - p_hat))

  # Granice dla przekształcenia logitowego
  q <- qnorm(1 - (alpha/2))

  part_logit = sqrt(1/(trials*p_hat*(1 - p_hat)))

  logit_lcl <- logit_p_hat - q * part_logit
  logit_ucl <- logit_p_hat + q * part_logit


  # Odwrócenie przekształcenia logitowego do p
  lcl <- exp(logit_lcl) / (1 + exp(logit_lcl))
  ucl <- exp(logit_ucl) / (1 + exp(logit_ucl))

  return(c(lower = lcl, upper = ucl))
}


alpha <- 0.05
conf_level <- 1 - alpha

trials <- 1000
sample_trial <- sample(c(1, 0), trials, replace = TRUE)
successes <- sum(sample_trial)

delta_confint <- logit_confint_delta(successes, trials, alpha)
binom_confint <- binom.confint(successes, trials, conf_level, methods = c
    ("exact", "asymptotic"))

cat("Metoda delta (logit):\n")
print(delta_confint)
cat("\nFunkcja binom.confint:\n")
print(binom_confint)
```

W wyniku działania powyższego kodu otrzymałyśmy przedziały ufności dla zaimplementowanej przez nas metody oraz dla funkcji wbudowanej.

Metoda	**Dolna granica**	**Górna granica**
Zaimplementowana metoda	0.4800152	0.5419005
Wbudowana - asymptotic	0.4800177	0.5419823
Wbudowana - exact	0.4795242	0.5424109

Tabela 20: Granice asymptotyczne przedziałów ufności

Jak możemy zauważyć w tabeli 20 granice asymptotyczne przedziałów ufności wyznaczone metodą zaimplementowaną przez nas oraz funkcją wbudowaną *binom.confint* są do

siebie bardzo zbliżone. Pozwala nam to wnioskować, że metoda zaimplementowana przez nas najprawdopodobniej jest poprawna.

8 Źródła

- Wykłady dr hab. inż. Krzysztofa Burneckiego oraz laboratoria dr inż. Aleksandry Grzesiek z przedmiotu „Statystyka stosowana”.
- Wykłady dr inż. Aleksandry Grzesiek oraz laboratoria inż. Huberta Woszczka z przedmiotu „Analiza danych ankietowych”.
- Wykłady dr inż. Andrzeja Giniewicza oraz laboratoria dr inż Agnieszki Kamińskiej z przedmiotu "Pakiety statystyczne".
- https://www.rdocumentation.org/
- https://www.r-project.org/other-docs.html