Metody przetwarzania i analizy danych w R

Łukasz Wawrowski

Contents

Literatura

Podstawowa:

- Przemysław Biecek - *Przewodnik po pakiecie R*
- Marek Gągolewski - *Programowanie w języku R. Analiza danych, obliczenia, symulacje.*
- Garret Grolemund, Hadley Wickham - *R for Data Science* (polska wersja)

Dodatkowa:

- inne pozycje po polsku
- inne pozycje po angielsku

Skrypty z zajęć

Chapter 1

Analiza opisowa

1.1 Wprowadzenie

Prezentacja

1.2 Eksploracyjna analiza danych

Explanatory Data Analysis (EDA) polega na opisie i wizualizacji danch bez zakładania hipotez badawczych. W R dostępnych jest wiele pakietów, które wspierają ten proces - The Landscape of R Packages for Automated Exploratory Data Analysis.

Najczęściej pobieramym pakietem jest summarytools z następującymi funkcjami:

- `view(dfSummary(zbior))` - wyświetla podsumowanie zawierające kilka statystyk opisowych, częstości, histogram i braki danych
- `freq(zbior$zmienna)` - wyświetla tabelę częstości
- `descr(zbior)` - wyświetla zestaw 15 statystyk opisowych

Poniżej znajduje się krótki opis tych 15 miar:

- średnia (mean) - przeciętna wartość cechy
- odchylenie standardowe (std. dev) - o ile wartości cechy odchylają się średnio od średniej
- minimum (min) - minimalna wartość cechy
- kwartyl pierwszy (Q1) - 25% obserwacji ma wartości poniżej Q1, a 75% obserwacji powyżej
- mediana (median) - wartość środkowa, 50% obserwacji ma wartości poniżej Q1, a 50% obserwacji powyżej

- kwartyl trzeci (Q3) - 75% obserwacji ma wartości poniżej Q3, a 25% obserwacji powyżej
- maksimum (max) - maksymalna wartość cechy
- odchylenie medianowe (MAD) - mediana odchyleń od mediany
- rozstęp międzykwartylowy (IQR) - różnica pomiędzy trzecim i pierwszym kwartylem
- współczynnik zmienności (CV) - udział odchylenia standardowego w średniej wartości cechy (w %)
- skośność (skewness) - asymetria rozkładu cechy, < 0 - lewostronna, > 0 - prawostronna, $= 0$ - symetryczny
- błąd standardowy skośności (se.skewness)
- kurtoza (kurtosis) - skupienie wartości wokół średniej, < 0 - słabe, > 0 - silne, $= 0$ - normalne
- liczba obserwacji bez braków danych (N.Valid)
- odsetek obserwacji bez braków danych (Pct.Valid)

Chapter 2

Testowanie hipotez

Prezentacja - raporty

Prezentacja - testy

2.1 Wprowadzenie

Do rozwiązania wybranych zagadnień analizy statystycznej wystarczą metody weryfikacji hipotez statystycznych. Taki proces można przedstawić w następujących krokach:

1. Sformułowanie dwóch wykluczających się hipotez - zerowej H_0 oraz alternatywnej H_1
2. Wybór odpowiedniego testu statystycznego
3. Określenie dopuszczalnego prawdopodobieństwo popełnienia błędu I rodzaju (czyli poziomu istotności α)
4. Podjęcie decyzji

Wymienione powyżej nowe pojęcia zostaną wyjaśnione poniżej.

2.2 Hipoteza statystyczna

Przypuszczenie dotyczące własności analizowanej cechy, np. średnia w populacji jest równa 10, rozkład cechy jest normalny.

Formułuje się zawsze dwie hipotezy: hipotezę zerową (H_0) i hipotezę alternatywną (H_1). Hipoteza zerowa jest hipotezą mówiącą o równości:

$H_0 : \bar{x} = 10$

Z kolei hipoteza alternatywna zakłada coś przeciwnego:

$H_1 : \bar{x} \neq 10$

Zamiast znaku nierówności ($\neq$) może się także pojawić znak mniejszości ($<$) lub większości ($>$).

2.3 Poziom istotności i wartość p

Hipotezy statystyczne weryfikuje się przy określonym poziomie istotności α, który wskazuje maksymalny poziom akceptowalnego błędu (najczęściej $\alpha = 0,05$).

Większość programów statystycznych podaje w wynikach testu wartość p. Jest to najostrzejszy poziom istotności, przy którym możemy odrzucić hipotezę H_0. Jest to rozwiązanie bardzo popularne, ale nie pozbawione wad. Dokładny opis potencjalnych zagrożeń można znaleźć w artykule.

Generalnie jeśli $p < \alpha$ - odrzucamy hipotezę zerową.

2.4 Testy statystyczne

W zależności od tego co chcemy weryfikować należy wybrać odpowiedni test. Tabela poniżej przedstawia dosyć wyczerpującą klasyfikację testów pobraną ze strony.

Overview of statistical tests

Type of dependent variable	Type of independent variable						
	Ordinal/categorical				Normal/interval (ordinal)	More than 1	None
	Two groups		More groups				
	Paired	Unpaired	Paired	Unpaired			
2 categories	McNemar Test, Sign-Test	Fisher Test, Chi-squared-Test	Cochran's Q-Test	Fisher Test, Chi-squared-test	(Conditional) Logistic Regression	Logistic Regression	Chi-squared-Test
Nominal	Bowker Test	Fisher Test, Chi-squared-Test		Fisher Test, Chi-squared-test	Multinomial logistic regression	Multinomial logistic regression	Binomial Test
Ordinal	Wilcoxon Test, Sign-Test	Wilcoxon-Mann-Whitney Test	Friedman-Test	Kruskal-Wallis Test	Spearman-rank-test	Ordered logit	Median Test
Interval	Wilcoxon Test, Sign-Test	Wilcoxon-Mann-Whitney Test	Friedman-Test	Kruskal-Wallis Test	Spearman-rank test	Multivariate linear model	Median Test
Normal	t-Test (for paired)	t-Test (for unpaired)	Linear Model (ANOVA)	Linear Model (ANOVA)	Pearson-Correlation-test	Multivariate Linear Model	t-Test
Censored Interval	Log-Rank Test		Survival Analysis, Cox proportional hazards regression				
None	Clustering, factor analysis, PCA, canonical correlation						

2.5 Zbiór danych

Będziemy działać na zbiorze danych dotyczącym pracowników przedsiębiorstwa. Poniżej znajduje się opis cech znajdujących się w tym zbiorze,

- id - kod pracownika
- plec - płeć pracownika (0 - mężczyzna, 1 - kobieta)
- data_urodz - data urodzenia
- edukacja - wykształcenie (w latach nauki)
- kat_pracownika - grupa pracownicza (1 - specjalista, 2 - menedżer, 3 - konsultant)
- bwynagrodzenie - bieżące wynagrodzenie
- pwynagrodzenie - początkowe wynagrodzenie
- staz - staż pracy (w miesiącach)
- doswiadczenie - poprzednie zatrudnienie (w miesiącach)
- zwiazki - przynależność do związków zawodowych (0 - nie, 1 - tak)
- wiek - wiek (w latach)

```
library(tidyverse)
library(readxl)

pracownicy <- read_excel("data/pracownicy.xlsx")
```

2.6 Test niezależności

Za pomocą testu niezależności χ^2 (chi-kwadrat) można sprawdzić czy pomiędzy dwiema cechami jakościowymi występuje zależność. Układ hipotez jest następujący:

- H_0 : zmienne są niezależne,
- H_1 : zmienne nie są niezależne.

W programie R test niezależności można wywołać za pomocą funkcji `chisq.test()` z pakietu *stats*. Jako argument tej funkcji należy podać tablicę kontyngencji. W przypadku operowania na danych jednostkowych można ją utworzyć poprzez funkcję `table()`. Jeżeli wprowadzamy liczebności ręcznie to należy zadbać o to, żeby wprowadzony obiekt był typu `matrix`.

Przykład

Czy pomiędzy zmienną płeć, a zmienną przynależność do związków zawodowych istnieje zależność?

W pierwszym kroku określamy hipotezy badawcze:

H_0: pomiędzy płcią a przynależnością do związków nie ma zależności

H_1: pomiędzy płcią a przynależnością do związków jest zależność

oraz przyjmujemy poziom istotności - weźmy standardową wartość $\alpha = 0,05$.

W pierwszej kolejności popatrzmy na tabelę krzyżową (kontyngencji) zawierającą liczebności poszczególnych kombinacji wariantów.

```
table(pracownicy$plec, pracownicy$zwiazki)
```

```
##
##       0   1
##   0 194  64
##   1 176  40
```

Wartości w tej tabeli nie wskazują na liczniejszą reprezentację jednej z płci w związkach zawodowych. Zweryfikujemy zatem wskazaną hipotezę zerową z wykorzystaniem testu χ^2.

```
chisq.test(table(pracownicy$plec, pracownicy$zwiazki))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(pracownicy$plec, pracownicy$zwiazki)
## X-squared = 2.3592, df = 1, p-value = 0.1245
```

Przy poziomie istotności $\alpha = 0,05$, wartości p (0.1245) jest większa od wartości α, zatem nie ma podstaw do odrzucenia hipotezy zerowej. Można stwierdzić, że nie ma zależności pomiędzy zmiennymi płeć i przynależność do związków zawodowych.

Przykład

Czy pomiędzy płcią, a grupami bieżącego wynagrodzenia zdefiniowanymi przez medianę istnieje zależność?

H_0: pomiędzy płcią a grupami wynagrodzenia nie ma zależności

H_1: pomiędzy płcią a grupami wynagrodzenia jest zależność

W pierwszej kolejności tworzymy nową cechą zamieniając cechę `bwynagrodzenie` na zmienną jakościową posiadającą dwa warianty: poniżej mediany i powyżej mediany.

```
pracownicy <- pracownicy %>%
  mutate(bwyn_mediana=cut(x = bwynagrodzenie,
                          breaks = c(min(bwynagrodzenie),
                                     median(bwynagrodzenie),
                                     max(bwynagrodzenie)),
                          include.lowest = TRUE))

table(pracownicy$plec, pracownicy$bwyn_mediana)
```

```
##
##     [1.58e+04,2.89e+04] (2.89e+04,1.35e+05]
##   0                  73                  185
##   1                 164                   52
```

W tym przypadku wygląd tablicy krzyżowej może sugerować występowanie zależności.

```
chisq.test(table(pracownicy$plec, pracownicy$bwyn_mediana))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(pracownicy$plec, pracownicy$bwyn_mediana)
## X-squared = 104.8, df = 1, p-value < 2.2e-16
```

Test χ^2 to potwierdza - mamy podstawy do odrzucenia hipotezy zerowej na korzyść hipotezy alternatywnej - istnieje zależność pomiędzy płcią, a grupami wynagrodzenia.

2.7 Test proporcji

Test proporcji pozwala odpowiedzieć na pytanie czy odsetki w jednej, dwóch lub więcej grupach różnią się od siebie istotnie. Dla jednej próby układ hipotez został przedstawiony poniżej:

- $H_0 : p = p_0$
- $H_1 : p \neq p_0$ lub $H_1 : p > p_0$ lub $H_1 : p < p_0$

Układ hipotez w przypadku dwóch prób jest następujący:

- $H_0 : p_1 = p_2$
- $H_1 : p_1 \neq p_2$ lub $H_1 : p_1 > p_2$ lub $H_1 : p_1 < p_2$

Dla k badanych prób hipotezę zerową i alternatywną można zapisać w następująco:

- $H_0 : p_1 = p_2 = p3 = ... = p_k$
- $H_1 : \exists\, p_i \neq p_j$

W takim przypadku hipoteza alternatywna oznacza, że co najmniej jeden odsetek różni się istotnie od pozostałych.

Funkcja `prop.test` z pakietu *stats* umożliwia przeprowadzanie testu proporcji w programie R. Jako argumenty należy podać wektor, który zawiera licznik badanych odsetków - `x`, oraz wektor zawierający wartości mianownika - `n`. W przypadku jednej próby należy jeszcze dodać argument `p`, którego wartość oznacza weryfikowany odsetek.

Przykład

Wysunięto przypuszczenie, że palacze papierosów stanowią jednakowy odsetek wśród mężczyzn i kobiet. W celu sprawdzenia tej hipotezy wylosowano 500 mężczyzn i 600 kobiet. Okazało się, że wśród mężczyzn było 200 palaczy, a wśród kobiet 250.

H_0: odsetek palaczy wg płci jest taki sam

H_1: odsetek palaczy różni się wg płci

```
prop.test(x = c(200,250), n = c(500,600))
```

```
##
##  2-sample test for equality of proportions with continuity correction
##
## data:  c(200, 250) out of c(500, 600)
## X-squared = 0.24824, df = 1, p-value = 0.6183
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.07680992  0.04347659
## sample estimates:
##    prop 1    prop 2
## 0.4000000 0.4166667
```

Przy poziomie istotności 0,05 nie ma podstaw do odrzucenia H0 - odsetek palaczy jest taki sam w grupach płci.

2.8 Testowanie normalności - test Shapiro-Wilka

Testy parametryczne z reguły wymagają spełnienia założenia o normalności rozkładu. W celu weryfikacji tego założenia należy wykorzystać jeden z testów normalności.

W celu formalnego zweryfikowania rozkładu cechy można wykorzystać test Shapiro-Wilka. Układ hipotez z tym teście jest następujący:

- $H_0 : F(x) = F_0(x)$ - rozkład cechy ma rozkład normalny
- $H_1 : F(x) \neq F_0(x)$ - rozkład cechy nie ma rozkładu normalnego

W przeprowadzonych dotychczas symulacjach wykazano, że test Shapiro-Wilka ma największą moc spośród testów normalności, niemniej jego ograniczeniem jest maksymalna liczba obserwacji, która wynosi 5000[1].

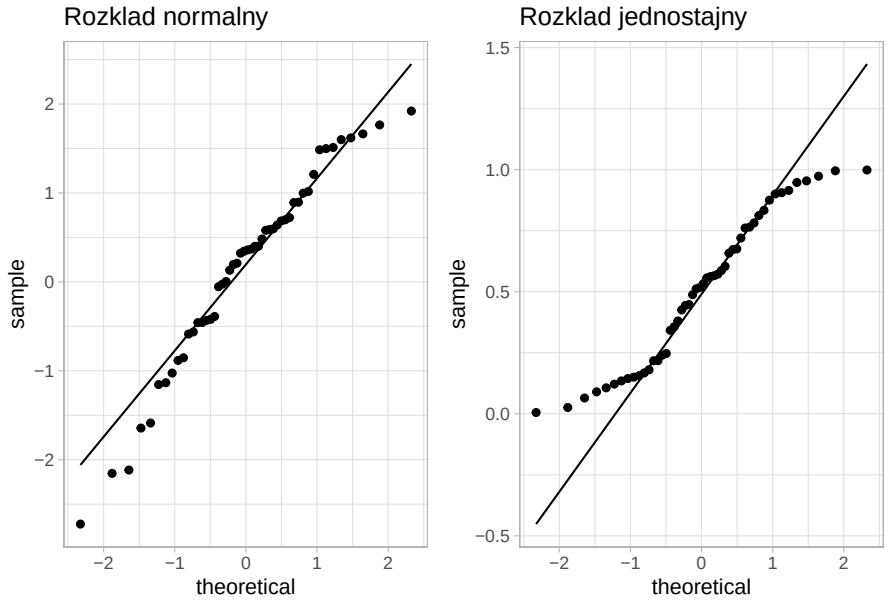
W programie R test Shapiro-Wilka można uruchomić za pomocą funkcji `shapiro.test()` jako argument podając wektor wartości liczbowych, który chcemy zweryfikować.

[1] W przypadku liczniejszych prób można wykorzystać test Kołmogorowa-Smirnova.

2.9 Testowanie normalności - wykres kwantyl-kwantyl

Normalność rozkładu może także zostać zweryfikowana poprzez utworzenie wykresu przedstawiającego porównanie wartości oryginalnych oraz odpowiadającym im wartości pochodzących z rozkładu normalnego. Dodatkowo prowadzona jest linia regresji pomiędzy otrzymanymi wartościami. Punkty przebiegające w pobliżu tej linii oznaczają, że rozkład tej cechy jest normalny.

Na wykresie przedstawiony jest wykres kwantyl-kwantyl dla 50 wartości wylosowanych z rozkładu normalnego i z rozkładu jednostajnego.



Jak można zauważyć punkty na wykresie po lewej stronie nie odbiegają znacząco od linii prostej, zatem można przypuszczać, że rozkład tej cechy jest normalny. Z kolei na wykresie po prawej stronie obserwuje się odstępstwo od rozkładu normalnego - wartości na krańcach linii są od niej oddalone.

Przykład

Czy cecha *doświadczenie* ma rozkład normalny? Sprawdź za pomocą odpowiedniego testu oraz wykresu kwantyl-kwantyl.

H_0: doświadczenie ma rozkład normalny

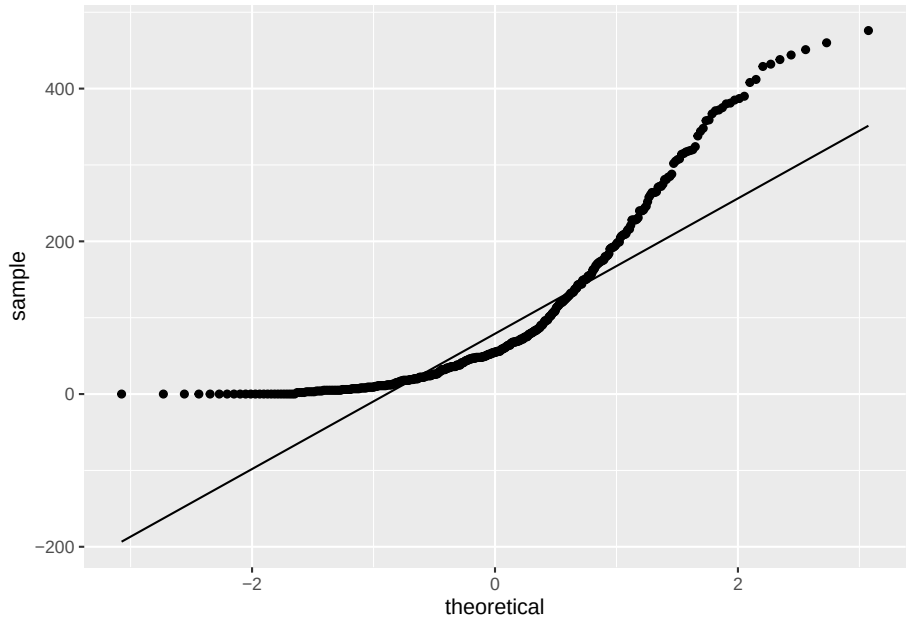
H_1: doświadczenie nie ma rozkładu normalnego

```
shapiro.test(pracownicy$doswiadczenie)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pracownicy$doswiadczenie
## W = 0.8136, p-value < 2.2e-16
```

Na poziomie $\alpha = 0,05$ Odrzucamy H_0 ($p < \alpha$) - doświadczenie nie ma rozkładu normalnego. Sprawdźmy jeszcze jak te wartości wyglądają na wykresie kwantyl-kwantyl.

```
ggplot(pracownicy, aes(sample = doswiadczenie)) +
  stat_qq() +
  stat_qq_line()
```



2.10 Testowanie wariancji - test Bartletta

Oprócz założenia o normalności, niektóre metody statystyczne wymagają także równości wariancji.

Jeśli chcemy sprawdzić homogeniczność wariancji w dwóch lub więcej grupach to należy skorzystać z testu Bartletta:

- $H_0 : s_1^2 = s_2^2 = s_3^2 = ... = s_k^2$
- $H_1 : \exists_{i,j \in \{1,..,k\}} \; s_i^2 \neq s_j^2$

Funkcja `bartlett.test()` w programie R umożliwia zastosowanie tego testu. Argumenty do tej funkcji można przekazać na dwa sposoby. Pierwszy polega

na przypisaniu do argumentu `x` wektora zawierającego wartości cechy, a do argumentu `g` wektora zawierającego identyfikatory poszczególnych grup. Drugi sposób to zadeklarowanie formuły w postaci `zmienna_analizowa ~ zmienna_grupująca` oraz podanie zbioru danych przypisanego do argumentu `data`.

Przykład

Sprawdźmy czy wariancje zmiennej `doświadczenie` w grupach płci są takie same.

H_0: wariancje doświadczenia są takie same w grupach płci

H_1: wariancje doświadczenia nie są takie same w grupach płci

Funkcję weryfikującą H_0 można zapisać na dwa sposoby - wynik zawsze będzie taki sam.

```
bartlett.test(x = pracownicy$doswiadczenie, g = pracownicy$plec)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  pracownicy$doswiadczenie and pracownicy$plec
## Bartlett's K-squared = 4.7659, df = 1, p-value = 0.02903
```

```
bartlett.test(pracownicy$doswiadczenie ~ pracownicy$plec)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  pracownicy$doswiadczenie by pracownicy$plec
## Bartlett's K-squared = 4.7659, df = 1, p-value = 0.02903
```

Przyjmując poziom istotności $\alpha = 0,05$ odrzucamy hipotezę zerową stwierdzając, że wariancje różnią się w grupach płci. Z kolei dopuszczając niższy poziom istotności $\alpha = 0,01$ podjęlibyśmy decyzję o braku podstaw do odrzucenia H_0 i nieistotnej różnicy pomiędzy grupami.

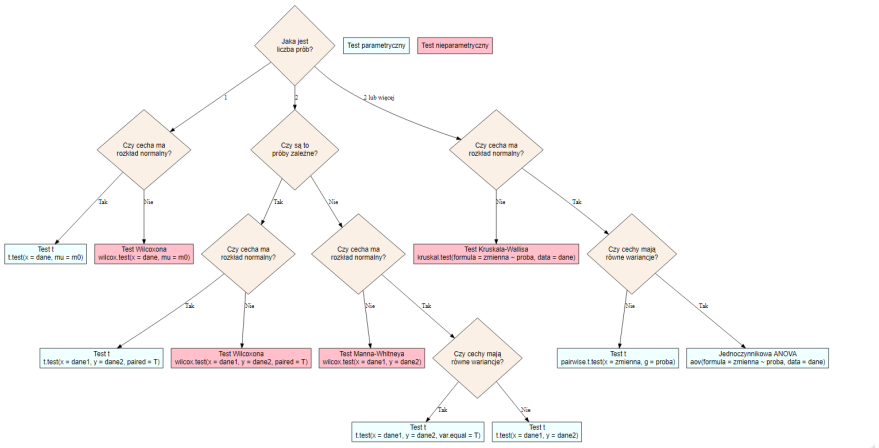
2.11 Testowanie średnich

W przypadku testowania wartości przeciętnych należy wprowadzić pojęcie prób zależnych i niezależnych:

- próby zależne (paired) - analizowane są te same jednostki, ale różne cechy.
- próby niezależne (unpaired) - analizowane są różne jednostki, ale ta sama cecha.

W zależności od tego czy spełnione są odpowiednie założenia dotyczące normalności cechy oraz równości wariancji należy wybrać odpowiedni test według

poniższego diagramu.



2.11.1 Test t-średnich

Weryfikacja równości średnich może odbywać się na zasadzie porównania wartości średniej w jednej grupie z arbitralnie przyjętym poziomem lub w dwóch różnych grupach. W pierwszym przypadku rozważamy układ hipotez:

- $H_0 : m = m_0$
- $H_1 : m \neq m_0$ lub $H_1 : m < m_0$ lub $H_1 : m > m_0$

natomiast w drugim przypadku hipotezy będą wyglądać następująco:

- $H_0 : m_1 = m_2$
- $H_1 : m_1 \neq m_2$ lub $H_1 : m_1 < m_2$ lub $H_1 : m_1 > m_2$

Alternatywnie hipotezę zerową można zapisać jako $m_1 - m_2 = 0$ czyli sprawdzamy czy różnica pomiędzy grupami istotnie różni się od zera.

W funkcji `t.test()` z pakietu *stats* w przypadku jednej próby należy podać argument `x` czyli wektor z wartościami, które są analizowane oraz wartość, z którą tą średnią porównujemy (argument `mu`, który domyślnie jest równy 0). Dodatkowo w argumencie `alternative` wskazujemy jaką hipotezę alternatywną bierzemy pod uwagę.

Dla weryfikacji równości średniej w dwóch próbach należy dodać argument `y` z wartościami w drugiej próbie. W tym przypadku mamy także możliwość określenia czy próby są zależne (argument `paired`) lub czy wariancja w obu próbach jest taka sama (`var.equal`). Jeżeli wariancje są różne to program R przeprowadzi test t Welcha i liczba stopni swobody nie będzie liczbą całkowitą.

2.11.2 ANOVA

W przypadku większej liczby grup stosuje się jednoczynnikową analizę wariancji (ANOVA). Ta analiza wymaga spełnienia założenia o normalności rozkładu i równości wariancji w badanych grupach. Układ hipotez jest następujący:

- $H_0 : m_1 = m_2 = m_3 = ... = m_k$
- $H_1 : \exists_{i,j \in \{1,..,k\}} \; m_i \neq m_j$

Za pomocą funkcji `aov()` można w R przeprowadzić jednoczynnikową analizę wariancji. Jako argument funkcji należy podać formułę przedstawiającą zależność zmiennej badanej do zmiennej grupującej wykorzystując w tym celu symbol tyldy (`~`) w następującym kontekście: `zmienna_analizowana ~ zmienna_grupująca`. Przy takim zapisie należy także w argumencie `data` podać nazwę zbioru danych.

W porównaniu do wcześniej opisanych funkcji, `aov()` nie zwraca w bezpośrednim wyniku wartości p. Aby uzyskać tę wartość należy wynik działania tej funkcji przypisać do obiektu, a następnie na nim wywołać funkcję `summary()`.

W przypadku odrzucenia hipotezy zerowej można przeprowadzić test Tukeya w celu identyfikacji różniących się par wykorzystując funkcję `TukeyHSD()` i jako argument podając obiekt zawierający wynik ANOVA.

W sytuacji, w której założenia użycia testu parametrycznego nie są spełnione, należy skorzystać z testów nieparametrycznych. W przypadku testowania miar tendencji centralnej różnica pomiędzy testami parametrycznymi a nieparametrycznymi polega na zastąpieniu wartości średniej medianą. Z punktu widzenia obliczeń w miejsce oryginalnych wartości cechy wprowadza się rangi czyli następuje osłabienie skali pomiarowej - z ilorazowej na porządkową.

2.11.3 Test Wilcoxona

Test Wilcoxona jest nieparametryczną wersją testu t. Hipotezy w tym teście dotyczą równości rozkładów:

- $H_0 : F_1 = F_2$
- $H_1 : F_1 \neq F_2$

Wartość statystyki testowej będzie zależna od typu testu, natomiast w R funkcja, której należy użyć to `wilcox.test()`. Argumenty tej funkcji są takie same jak w przypadku testu t.

2.11.4 Test Kruskala-Wallisa

Z kolei test Kruskala-Wallisa jest nieparametrycznym odpowiednikiem ANOVA. Hipotezy są następujące:

- $H_0 : F_1 = F_2 = F_3 = ... = F_k$
- $H_1 : \exists_{i,j \in \{1,..,k\}} \; F_i \neq F_j$

W programie R korzysta się z funkcji `kruskal.test()`, która przyjmuje takie same argumenty jak funkcja do metody ANOVA `aov()`. Główną różnicą jest sposób podawania wyniku testu, ponieważ w tym przypadku od razu otrzymujemy wartość p. W przypadku odrzucenia hipotezy zerowej należy sprawdzić, które grupy różnią się między sobą. Można to zrobić za pomocą funkcji `pairwise.wilcox.test()`.

Przykład

Sprawdzimy czy średnie doświadczenie w grupach płci jest takie same.

H_0: średnie doświadczenie w grupach płci jest takie samo

H_1: średnie doświadczenie w grupach płci nie jest takie samo

W związku z tym, że badana cecha nie ma rozkładu normalnego zostanie przeprowadzony test Wilcoxona. Mamy tutaj do czynienia z testem dla prób niezależnych - badana jest jedna cecha (doświadczenie) w ramach rozłącznych grup płci.

```
wilcox.test(pracownicy$doswiadczenie ~ pracownicy$plec)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  pracownicy$doswiadczenie by pracownicy$plec
## W = 36295, p-value = 0.00000001372
## alternative hypothesis: true location shift is not equal to 0
```

Przyjmując poziom istotności $\alpha = 0,05$ odrzucamy H_0 - średnie doświadczenie nie jest takie samo.

Przykład

Czy początkowe i bieżące wynagrodzenie różni się od siebie w sposób istotny?

H_0: średnie początkowe i bieżące wynagrodzenie jest takie samo

H_1: średnie początkowe i bieżące wynagrodzenie nie jest takie samo

W pierwszej kolejności weryfikujemy normalność rozkładu analizowanych cech.

```
shapiro.test(pracownicy$pwynagrodzenie)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pracownicy$pwynagrodzenie
## W = 0.71535, p-value < 2.2e-16
```

```
shapiro.test(pracownicy$bwynagrodzenie)
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  pracownicy$bwynagrodzenie
## W = 0.77061, p-value < 2.2e-16
```

Wynagrodzenie w tym zbiorze danych zdecydowanie nie przypomina rozkładu normalnego. W tym przypadku analizujemy próby zależne - badamy dwie różne cechy dla tych samych jednostek (obserwacji).

```
wilcox.test(x = pracownicy$pwynagrodzenie,
            y = pracownicy$bwynagrodzenie,
            paired = TRUE)
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  pracownicy$pwynagrodzenie and pracownicy$bwynagrodzenie
## V = 0, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0
```

Na podstawie podanej wartości p odrzucamy H_0 - średnie początkowe i bieżące wynagrodzenie różni się od siebie istotnie statystycznie.

Przykład

Analogicznie można także sprawdzić czy np. doświadczenie różni się w ramach więcej niż dwóch grup - w takim przypadku rozpatrujemy głównie próby niezależne.

H_0: średnie doświadczenie w grupach kategorii pracownika jest takie same

H_1: średnie doświadczenie w grupach kategorii pracownika nie jest takie same - co najmniej jedna para jest różna

```
kruskal.test(pracownicy$doswiadczenie ~ pracownicy$kat_pracownika)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  pracownicy$doswiadczenie by pracownicy$kat_pracownika
## Kruskal-Wallis chi-squared = 57.466, df = 2, p-value = 3.322e-13
```

Przyjmując poziom istotności $\alpha = 0,05$ odrzucamy hipotezę zerową - co najmniej jedna para kategorii pracownika różni się pod względem średniego wynagrodzenia.

Chapter 3

Regresja

Prezentacja

3.1 Wprowadzenie

Metody regresji pozwalają na analizowanie zależności przyczynowo-skutkowych oraz predycję nieznanych wartości. Korzystając z tej metody należy jednak pamiętać, że model regresji jest tylko przybliżeniem rzeczywistości.

Przykłady zastosowania regresji:

- zależność wielkości sprzedaży od wydatków na reklamę
- zależność wynagrodzenia od lat doświadczenia

Na początku pracy wczytujemy biblioteki *tidyverse* i *readxl*.

```
library(tidyverse)
library(readxl)
```

3.2 Regresja prosta

Na podstawie danych dotyczących informacji o doświadczeniu i wynagrodzeniu pracowników zbuduj model określający 'widełki' dla potencjalnych pracowników o doświadczeniu równym 8, 10 i 11 lat.

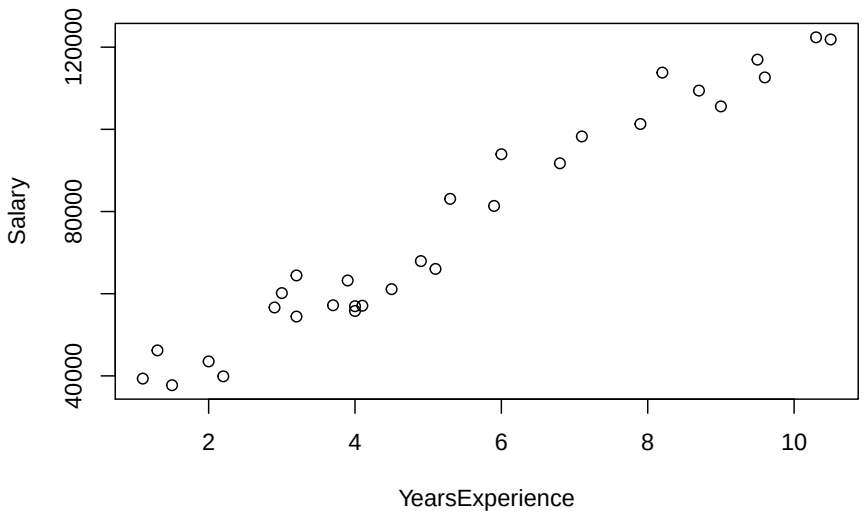
Wczytujemy dane i sprawdzamy czy nie występują zera bądź braki danych z użyciem funkcji `summary()`.

```
salary <- read_xlsx("data/salary.xlsx")

summary(salary)
```

```
##  YearsExperience      Salary
##  Min.   : 1.100   Min.   : 37731
##  1st Qu.: 3.200   1st Qu.: 56721
##  Median : 4.700   Median : 65237
##  Mean   : 5.313   Mean   : 76003
##  3rd Qu.: 7.700   3rd Qu.:100545
##  Max.   :10.500   Max.   :122391
```

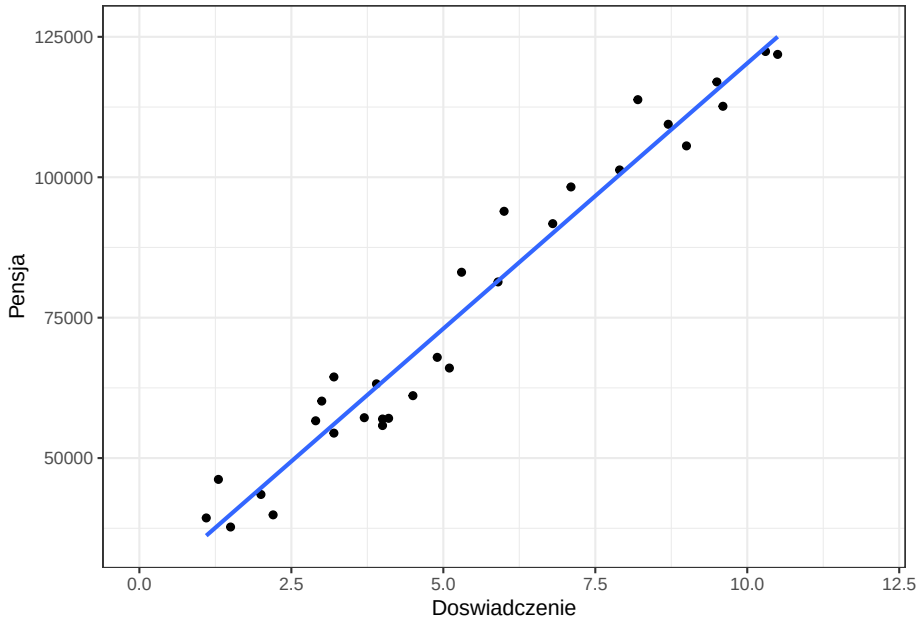
Następnie stworzymy wykres.

```
plot(salary)
```



Najprostszym sposobem wizualizacji jest wykorzystanie funkcji `plot()`, niemniej taki wykres nie jest najpiękniejszy i trudno się go formatuje. Dużo lepiej skorzystać z pakietu `ggplot2`.

```
ggplot(salary, aes(x=YearsExperience, y=Salary)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  xlab("Doświadczenie") +
  ylab("Pensja") +
  xlim(0,12) +
  ylim(35000, 126000) +
  theme_bw()
```



W argumentach funkcji `ggplot()` podajemy co wizualizujemy, natomiast sposób prezentacji ustalany jest przez funkcje `geom`. Funkcje `xlab()` i `ylab()` określają etykiety osi, a `xlim()` i `ylim()` wartości graniczne. Funkcje rozpoczynające się od `theme_` określają wygląd wykresu.

Modelowanie rozpoczynamy od określenia zmiennej zależnej i niezależnej.

- zmienna zależna/objaśniana: pensja - y
- zmienna niezależna/objaśniająca: doświadczenie - x

Szukamy teraz wzoru na prostą opisującą badane cechy.

Ogólna postać regresji prostej jest następująca:

$$\hat{y}_i = b_1 x_i + b_0$$

gdzie $\hat{y}$ oznacza wartość teoretyczną, leżącą na wyznaczonej prostej, x_i wartość zmiennej niezależnej, a b_1 i b_0 to współczynniki modelu.

Wobec tego wartości empiryczne y będą opisane formułą:

$$y_i = b_1 x_i + b_0 + u_i$$

w której u_i oznacza składnik resztowy wyliczany jako $u_i = y_i - \hat{y}_i$.

Współczynnik kierunkowy b_1 informuje o ile przeciętne zmieni się wartość zmiennej objaśnianej y, gdy wartość zmiennej objaśniającej x wzrośnie o jednostkę.

Wyraz wolny b_0 to wartość zmiennej objaśnianej y, w sytuacji w której wartość zmiennej objaśniającej x będzie równa 0. Często interpretacja tego parametru nie ma sensu.

Następnie w R korzystamy z funkcji `lm`, w której należy określić zależność funkcyjną w formie: `zmienna_zalezna ~ zmienna niezalezna`.

```
salary_model <- lm(formula = Salary ~ YearsExperience, data = salary)
summary(salary_model)
```

```
##
## Call:
## lm(formula = Salary ~ YearsExperience, data = salary)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7958.0 -4088.5  -459.9  3372.6 11448.0
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      25792.2     2273.1   11.35 5.51e-12 ***
## YearsExperience   9450.0      378.8   24.95  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5788 on 28 degrees of freedom
## Multiple R-squared:  0.957,  Adjusted R-squared:  0.9554
## F-statistic: 622.5 on 1 and 28 DF,  p-value: < 2.2e-16
```

Na podstawie otrzymanego wyniku dokonujemy interpretacji parametrów.

- b1 = 9450 - wzrost doświadczenia o rok powoduje przeciętny wzrost pensji o 9450 $
- b0 = 25792,2 - pracownik o doświadczeniu 0 lat uzyska pensję w wysokości 25792,2 $

Trzy gwiazki przy współczynniku b_1 oznaczają, że doświadczenie ma istotny wpływ na wartości pensji. Wyraz wolny także jest istotny, natomiast ogólnie nie jest wymagana jesgo istotność.

Oprócz interpretacji współczynników ważne są jeszcze inne miary jakości modelu.

Odchylenie standardowe składnika resztowego jest pierwiastkiem z sumy kwadratów reszt podzielonej przez liczbę obserwacji pomniejszoną o 2:

$$S_u = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}}$$

Błąd resztowy (odchylenie standardowe składnika resztowego) S_u wynosi 5788$, co oznacza, że wartości obliczone na podstawie modelu różnią się od rzeczywistości średnio o +/- 5788 $

Współczynnik determinacji określa, jaki procent wariancji zmiennej objaśnianej został wyjaśniony przez funkcję regresji. R^2 przyjmuje wartości z przedziału $< 0; 1 >$ ($< 0\%; 100\% >$), przy czym model regresji tym lepiej opisuje zachowanie się badanej zmiennej objaśnianej, im R^2 jest bliższy jedności (bliższy 100%)

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2}$$

Współczynnik R^2 (multiple R-squared) wynosi 0,957 czyli doświadczenie wyjaśnia 95,7% zmienności pensji

Na podstawie tego modelu dokonamy wyznaczenia wartości teoretycznych dla kilku wybranych wartości doświadczenia. Nowe wartości muszą być w postaci zbioru danych, zatem tworzymy nową ramkę danych.

```
nowiPracownicy <- data.frame(YearsExperience=c(8,10,11))

predict(salary_model, nowiPracownicy)
```

```
##        1        2        3
## 101391.9 120291.8 129741.8
```

Tym sposobem uzyskujemy następujące widełki:

- pracownik o 8 latach doświadczenia - proponowana pensja 101391,9 $ +/- 5788 $
- pracownik o 10 latach doświadczenia - proponowana pensja 120291,8 $ +/- 5788 $
- pracownik o 11 latach doświadczenia - proponowana pensja 129741,8 $ +/- 5788 $

W powyższym przykładzie prognozowane wartości pojawiły się w konsoli, natomiast w sytuacji, w której chcielibyśmy wykonać predykcję dla bardzo wielu nowych wartości to warto te prognozowane wartości umieścić od razu w zbiorze danych. Można to zrobić w następujący sposób:

```
nowiPracownicy <- nowiPracownicy %>%
  mutate(salary_pred=predict(object = salary_model, newdata = .))

nowiPracownicy
```

```
##   YearsExperience salary_pred
## 1               8    101391.9
## 2              10    120291.8
## 3              11    129741.8
```

W powyższym kodzie symbol . oznacza analizowany zbiór danych, zatem nie ma potrzeby powtarzania jego nazwy.

3.2.1 Zadanie

Dla danych dotyczących sklepu nr 77 opracuj model zależności sprzedaży od liczby klientów. Ile wynosi teoretyczna sprzedaż w dniach, w których liczba klientów będzie wynosiła 560, 740, 811 oraz 999 osób?

3.3 Regresja wieloraka

Ogólna postać regresji wielorakiej jest następująca:

$$\hat{y}_i = b_1 x_{1i} + b_2 x_{2i} + ... + b_k x_{ki} + b_0$$

W tym przypadku nie wyznaczamy prostej tylko k-wymiarową przestrzeń.

Na podstawie danych dotyczących zatrudnienia opracuj model, w którym zmienną zależną jest bieżące wynagrodzenie. Za pomocą regresji określimy jakie cechy mają istotny wpływ na to wynagrodzenie.

Opis zbioru:

- id - kod pracownika
- plec - płeć pracownika (0 - mężczyzna, 1 - kobieta)
- data_urodz - data urodzenia
- edukacja - wykształcenie (w latach nauki)
- kat_pracownika - grupa pracownicza (1 - specjalista, 2 - menedżer, 3 - konsultant)
- bwynagrodzenie - bieżące wynagrodzenie
- pwynagrodzenie - początkowe wynagrodzenie
- staz - staż pracy (w miesiącach)
- doswiadczenie - poprzednie zatrudnienie (w miesiącach)
- zwiazki - przynależność do związków zawodowych (0 - nie, 1 - tak)
- wiek - wiek (w latach)

Rozpoczynamy od wczytania danych,

```
pracownicy <- read_xlsx("data/pracownicy.xlsx")

pracownicy2 <- pracownicy %>%
  filter(!is.na(wiek)) %>%
  select(-id, -data_urodz) %>%
  mutate(plec=as.factor(plec),
         kat_pracownika=as.factor(kat_pracownika),
         zwiazki=as.factor(zwiazki))

summary(pracownicy2)
```

```
##  plec       edukacja     kat_pracownika bwynagrodzenie   pwynagrodzenie
##  0:257   Min.   : 8.00   1:362          Min.   : 15750   Min.   : 9000
##  1:216   1st Qu.:12.00   2: 27          1st Qu.: 24000   1st Qu.:12450
##          Median :12.00   3: 84          Median : 28800   Median :15000
##          Mean   :13.49                  Mean   : 34418   Mean   :17009
##          3rd Qu.:15.00                  3rd Qu.: 37050   3rd Qu.:17490
##          Max.   :21.00                  Max.   :135000   Max.   :79980
##       staz       doswiadczenie    zwiazki      wiek
##  Min.   :63.00   Min.   :  0.00   0:369   Min.   :24.00
##  1st Qu.:72.00   1st Qu.: 19.00   1:104   1st Qu.:30.00
##  Median :81.00   Median : 55.00           Median :33.00
##  Mean   :81.14   Mean   : 95.95           Mean   :38.67
##  3rd Qu.:90.00   3rd Qu.:139.00           3rd Qu.:47.00
##  Max.   :98.00   Max.   :476.00           Max.   :66.00
```

W zmiennej wiek występował brak danych, który został usunięty. Usunięto także kolumny, które nie przydadzą się w modelowaniu. Ponadto dokonujemy przekształcenia typu cech, które są jakościowe (płeć, kat_pracownika, zwiazki) z typu liczbowego na czynnik/faktor. Taka modyfikacja powoduje, że ta cecha będzie przez model traktowana jako zmienna dychotomiczna (zerojedynkowa). Proces trasformacji takiej cechy jest pokazany poniżej.

Oryginalny zbiór

id	stanowisko
1	specjalista
2	menedżer
3	specjalista
4	konsultant
5	konsultant

Zmienna zerojedynkowa

id	menedżer	konsultant
1	0	0
2	1	0
3	0	0
4	0	1
5	0	1

W modelu zmienna zależna to `bwynagrodzenie`, natomiast jako zmienne niezależne bierzemy pod uwagę wszystkie pozostałe cechy. W celu uniknięcia notacji naukowej w uzyskiwanych wynikach dodajemy opcję `options(scipen = 5)`.

```
options(scipen = 5)

model <- lm(bwynagrodzenie ~ ., pracownicy2)
summary(model)
```

```
##
## Call:
## lm(formula = bwynagrodzenie ~ ., data = pracownicy2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -23185  -3041   -705   2591  46295
##
## Coefficients:
##                    Estimate  Std. Error t value Pr(>|t|)
## (Intercept)     -4764.87418  3590.49652  -1.327  0.18514
## plec1           -1702.43743   796.51779  -2.137  0.03309 *
## edukacja          482.43603   160.83977   2.999  0.00285 **
## kat_pracownika2  6643.17910  1638.06138   4.056 5.87e-05 ***
## kat_pracownika3 11169.64519  1372.73990   8.137 3.77e-15 ***
## pwynagrodzenie      1.34021     0.07317  18.315  < 2e-16 ***
## staz              154.50876    31.65933   4.880 1.46e-06 ***
## doswiadczenie     -15.77375     5.78369  -2.727  0.00663 **
## zwiazki1        -1011.55276   787.80884  -1.284  0.19978
## wiek              -64.78787    47.88015  -1.353  0.17668
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6809 on 463 degrees of freedom
## Multiple R-squared:  0.8444, Adjusted R-squared:  0.8414
## F-statistic: 279.1 on 9 and 463 DF,  p-value: < 2.2e-16
```

Tak zbudowany model wyjaśnia 84,4% zmienności bieżącego wynagrodzenia, ale nie wszystkie zmienne są w tym modelu istotne.

Parametry regresji mają następujące interpretacje:

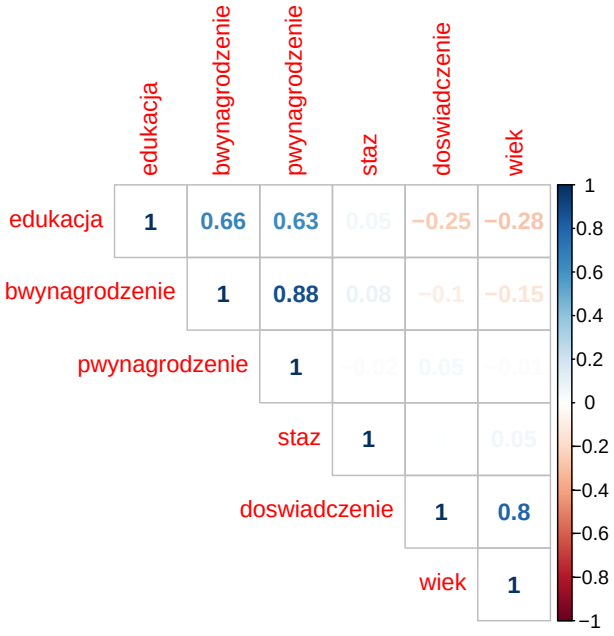
- plec1 - kobiety zarabiają przeciętnie o 1702,44 zł mniej niż mężczyźni,
- edukacja - wzrost liczby lat nauki o rok powoduje średni wzrost bieżącego wynagrodzenia o 482,44 zł
- kat_pracownika2 - pracownicy o kodzie 2 (menedżer) zarabiają średnio o 6643,18 zł więcej niż pracownik o kodzie 1 (specjalista)
- kat_pracownika2 - pracownicy o kodzie 3 (konsultant) zarabiają średnio o 11169,65 zł więcej niż pracownik o kodzie 1 (specjalista)
- pwynagrodzenie - wzrost początkowego wynagrodzenia o 1 zł powoduje przeciętny wzrost bieżącego wynagrodzenia o 1,34 zł
- staz - wzrost stażu pracy o miesiąc skutkuje przeciętnym wzrostem bieżącego wynagrodzenia o 154,51 zł
- doswiadcznie - wzrost doświadczenia o miesiąc powoduje średni spadek bieżącego wynagrodzenia o 15,77 zł
- zwiazki1 - pracownicy należący do związków zawodowych zarabiają średnio o 1011,55 zł mniej aniżeli pracownicy, którzy do związków nie zależą
- wiek - wzrost wieku pracownika o 1 rok to przeciętnym spadek bieżącego wynagrodzenia o 64,79 zł

Wszystkie te interpretacje obowiązują przy założeniu *ceteris paribus* - przy pozostałych warunkach niezmienionych.

Ten model wymaga oczywiście ulepszenia do czego wykorzystamy pakiet olsrr.

Pierwszą kwestią, którą się zajmiemy jest współliniowość zmiennych. W regresji zmienne objaśniające powinny być jak najbardziej skorelowane ze zmienną objaśnianą, a możliwie nieskorelowane ze sobą. W związku z tym wybieramy ze zbioru wyłącznie cechy ilościowe, dla którym wyznaczymy współczynnik korelacji liniowej Pearsona. Do wizualizacji tych wartości wykorzystamy pakiet corrplot służący do wizualizacji współczynnika korelacji.

```
library(corrplot)
library(olsrr)

korelacje <- pracownicy2 %>%
  select(-c(plec, kat_pracownika, zwiazki)) %>%
  cor()

corrplot(korelacje, method = "number", type = "upper")
```



Możemy zauważyć, że wartości bieżącego wynagrodzenia są najsilniej skorelowane w wartościami wynagrodzenia początkowego. Także doświadczenie i wiek są silnie ze sobą związane, co może sugerować, że obie zmienne wnoszą do modelu podobną informację.

W związku z tym powinniśmy wyeliminować niektóre zmienne z modelu pozostawiając te najważniejsze. Wyróżnia się trzy podejścia do tego zagadnienia:

- ekspercki dobór cech,
- budowa wszystkich możliwych modeli i wybór najlepszego według określonego kryterium,
- regresja krokowa.

W przypadku budowy wszystkich możliwych modeli należy pamiętać o rosnącej wykładniczo liczbie modeli: $2^p - 1$, gdzie p oznacza liczbę zmiennych objaśniających. w rozważanym przypadku liczba modeli wynosi 255.

```
wszystkie_modele <- ols_step_all_possible(model)
```

W uzyskanym zbiorze danych są informacje o numerze modelu, liczbie użytych zmiennych, nazwie tych zmiennych oraz wiele miar jakości. Te, które warto wziąć pod uwagę to przede wszystkim:

- `rsquare` - współczynnik R-kwadrat,
- `aic` - kryterium informacyjne Akaike,
- `msep` - błąd średniokwadratowy predykcji.

Najwyższa wartość współczynnika R^2 związana jest z modelem zawierającym

wszystkie dostępne zmienne objaśniające. Jest to pewna niedoskonałość tej miary, która rośnie wraz z liczbą zmiennych w modelu, nawet jeśli te zmienne nie są istotne.

W przypadku kryteriów informacyjnych oraz błędu średniokwadratowego interesują nas jak najmniejsze wartości. Wówczas jako najlepszy należy wskazać model nr 219 zawierający 6 zmiennych objaśniających.

Metodą, która także pozwoli uzyskać optymalny model, ale przy mniejszym obciążeniu obliczeniowym jest regresja krokowa polegająca na krokowym budowaniu modelu.

```
ols_step_both_aic(model)
```

```
##
##
##                                        Stepwise Summary
## ------------------------------------------------------------------------------------------
## Variable            Method       AIC             RSS               Sum Sq          R-Sq      Ad
## ------------------------------------------------------------------------------------------
## pwynagrodzenie    addition    9862.260    31053506813.535    106862706669.340    0.77484
## kat_pracownika    addition    9786.152    26215474648.689    111700738834.186    0.80992
## doswiadczenie     addition    9743.487    23853248017.651    114062965465.224    0.82705
## staz              addition    9719.469    22576592070.620    115339621412.255    0.83630
## edukacja          addition    9707.338    21912088629.084    116004124853.791    0.84112
## plec              addition    9703.188    21629051655.016    116287161827.859    0.84317
## ------------------------------------------------------------------------------------------
```

Otrzymany w ten sposób model jest tożsamy z modelem charakteryzującym się najlepszymi miarami jakości spośród zbioru wszystkich możliwych modeli:

```
wybrany_model <- lm(bwynagrodzenie ~ pwynagrodzenie + kat_pracownika + doswiadczenie + staz + ple
summary(wybrany_model)
```

```
##
## Call:
## lm(formula = bwynagrodzenie ~ pwynagrodzenie + kat_pracownika +
##     doswiadczenie + staz + plec + edukacja, data = pracownicy2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -22922  -3300   -673   2537  46524
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -6547.147   3402.860  -1.924  0.05496 .
## pwynagrodzenie      1.342      0.073  18.382  < 2e-16 ***
## kat_pracownika2  6734.992   1631.122   4.129 4.32e-05 ***
```

```
## kat_pracownika3 11226.635   1368.413   8.204 2.30e-15 ***
## doswiadczenie     -22.302      3.571  -6.246 9.51e-10 ***
## staz              147.865     31.461   4.700 3.43e-06 ***
## plec1           -1878.949    761.703  -2.467  0.01399 *
## edukacja          501.391    160.270   3.128  0.00187 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6820 on 465 degrees of freedom
## Multiple R-squared:  0.8432, Adjusted R-squared:  0.8408
## F-statistic: 357.1 on 7 and 465 DF,  p-value: < 2.2e-16
```

Uzyskany model charakteryzuje się nieznacznie większym błędem resztowym od modelu ze wszystkimi zmiennymi, ale wszystkie zmienne są istotne. Wyraz wolny `(Intercept)` nie musi być istotny w modelu.

Wróćmy jeszcze na chwilę do tematu współliniowości zmiennych objaśniających:

```
ols_vif_tol(wybrany_model)
```

```
##         Variables Tolerance      VIF
## 1  pwynagrodzenie 0.2979792 3.355939
## 2 kat_pracownika2 0.6867111 1.456217
## 3 kat_pracownika3 0.3595692 2.781106
## 4   doswiadczenie 0.7054176 1.417600
## 5            staz 0.9862198 1.013973
## 6           plec1 0.6831049 1.463904
## 7        edukacja 0.4607529 2.170361
```

Współczynnik tolerancji wskazuje na procent niewyjaśnionej zmienności danej zmiennej przez pozostałe zmienne objaśniające. Przykładowo współczynnik tolerancji dla początkowego wynagrodzenia wynosi 0,2980, co oznacza, że 30% zmienności początkowego wynagrodzenia nie jest wyjaśnione przez pozostałe zmienne w modelu. Z kolei współczynnik VIF jest obliczany na podstawie wartości współczynnika tolerancji i wskazuje o ile wariancja szacowanego współczynnika regresji jest podwyższona z powodu współliniowości danej zmiennej objaśniającej z pozostałymi zmiennymi objaśniającymi. Wartość współczynnika VIF powyżej 4 należy uznać za wskazującą na współliniowość. W analizowanym przypadku takie zmienne nie występują.

Ocena siły wpływu poszczególnych zmiennych objaśniających na zmienną objaśnianą w oryginalnej postaci modelu nie jest możliwa. Należy wyznaczyć standaryzowane współczynniki beta, które wyliczane są na danych standaryzowanych, czyli takich, które są pozbawione jednostek i cechują się średnią równą 0, a odchyleniem standardowym równym 1. Standaryzacja ma sens tylko dla cech numerycznych, w związku z czym korzystamy z funkcji `mutate_if()`, która jako pierwszy argument przyjmuje warunek, który ma być spełniony, aby była zastosowane przekształcenie podawane jako drugi argument.

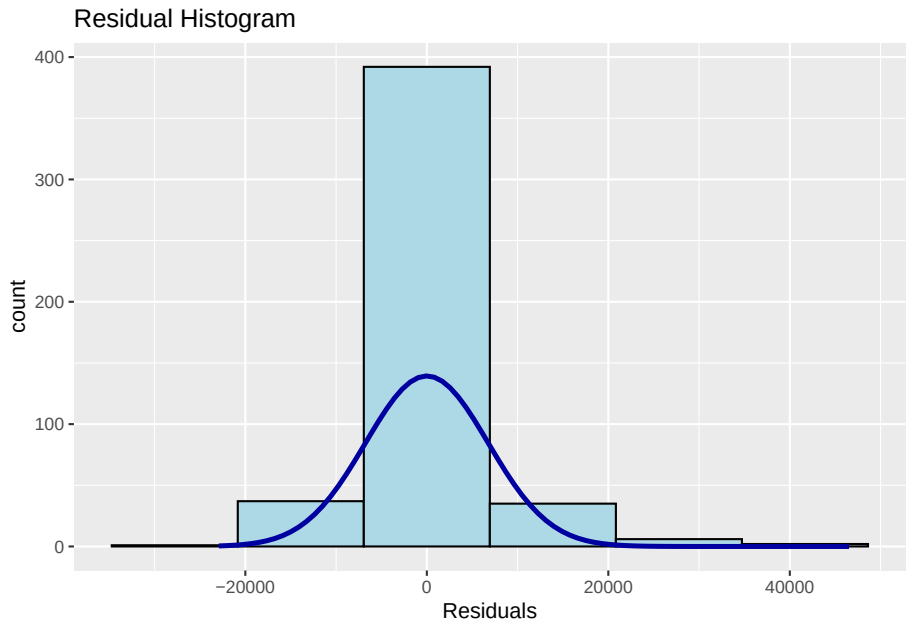
```
pracownicy2_std <- pracownicy2 %>%
  mutate_if(is.numeric, scale)

wybrany_model_std <- lm(bwynagrodzenie ~ pwynagrodzenie + kat_pracownika +
                          doswiadczenie + staz + plec + edukacja, data = pracownicy2_std)
summary(wybrany_model_std)
```

```
##
## Call:
## lm(formula = bwynagrodzenie ~ pwynagrodzenie + kat_pracownika +
##     doswiadczenie + staz + plec + edukacja, data = pracownicy2_std)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.34098 -0.19306 -0.03939  0.14841  2.72171
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -0.08893    0.03144  -2.828  0.00488 **
## pwynagrodzenie   0.61842    0.03364  18.382  < 2e-16 ***
## kat_pracownika2  0.39400    0.09542   4.129 4.32e-05 ***
## kat_pracownika3  0.65677    0.08005   8.204 2.30e-15 ***
## doswiadczenie   -0.13657    0.02187  -6.246 9.51e-10 ***
## staz             0.08691    0.01849   4.700 3.43e-06 ***
## plec1           -0.10992    0.04456  -2.467  0.01399 *
## edukacja         0.08464    0.02706   3.128  0.00187 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.399 on 465 degrees of freedom
## Multiple R-squared:  0.8432, Adjusted R-squared:  0.8408
## F-statistic: 357.1 on 7 and 465 DF,  p-value: < 2.2e-16
```

Spośród cech ilościowych największy wpływ na zmienną objaśnianą mają wartości wynagrodzenia początkowego, staż, edukacja i na końcu doświadczenie.

Reszty czyli różnice pomiędzy obserwowanymi wartościami zmiennej objaśnianej, a wartościami wynikającymi z modelu w klasycznej metodzie najmniejszych kwadratów powinny być zbliżone do rozkładu normalnego. Oznacza to, że najwięcej reszt powinno skupiać się wokół zerowych różnic, natomiast jak najmniej powinno być wartości modelowych znacznie różniących się od tych rzeczywistych.

```
ols_plot_resid_hist(wybrany_model)
```



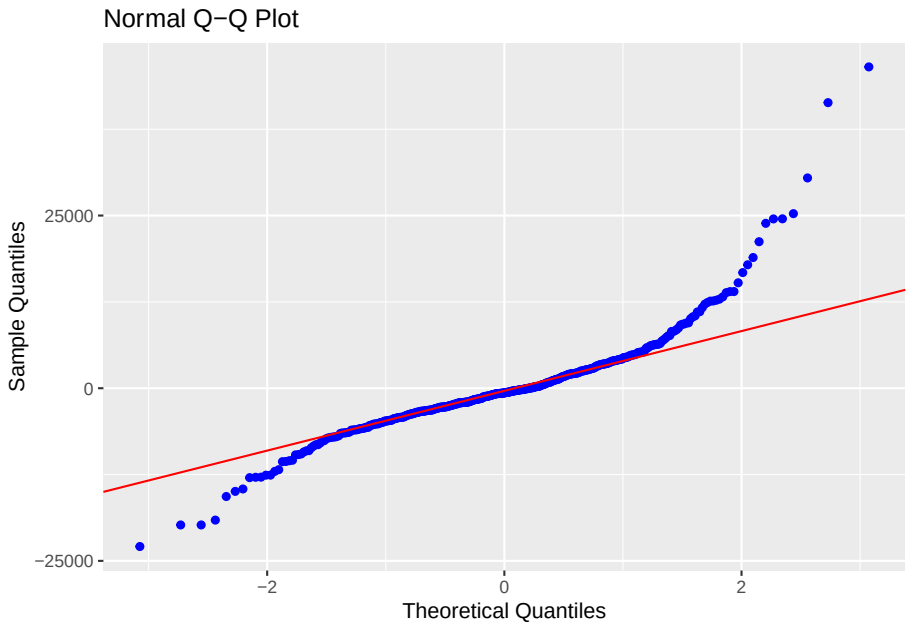
Reszty w naszym modelu wydają się być zbliżone do rozkładu normalnego. Jednoznaczną odpowiedź da jednak odpowiedni test.

```
ols_test_normality(wybrany_model)
```

```
## -----------------------------------------------
##        Test             Statistic       pvalue
## -----------------------------------------------
## Shapiro-Wilk              0.868          0.0000
## Kolmogorov-Smirnov        0.1092         0.0000
## Cramer-von Mises         42.5504         0.0000
## Anderson-Darling         13.0233         0.0000
## -----------------------------------------------
```

Hipoteza zerowa w tych testach mówi o zgodności rozkładu reszt z rozkładem normalnym. Na podstawie wartości p, które są mniejsze od $\alpha = 0,05$ stwierdzamy, że są podstawy do odrzucenia tej hipotezy czyli reszty z naszego modelu nie mają rozkładu normalnego. W diagnostyce przyczyn takiego stanu rzeczy pomoże nam wykres kwantyl-kwantyl:

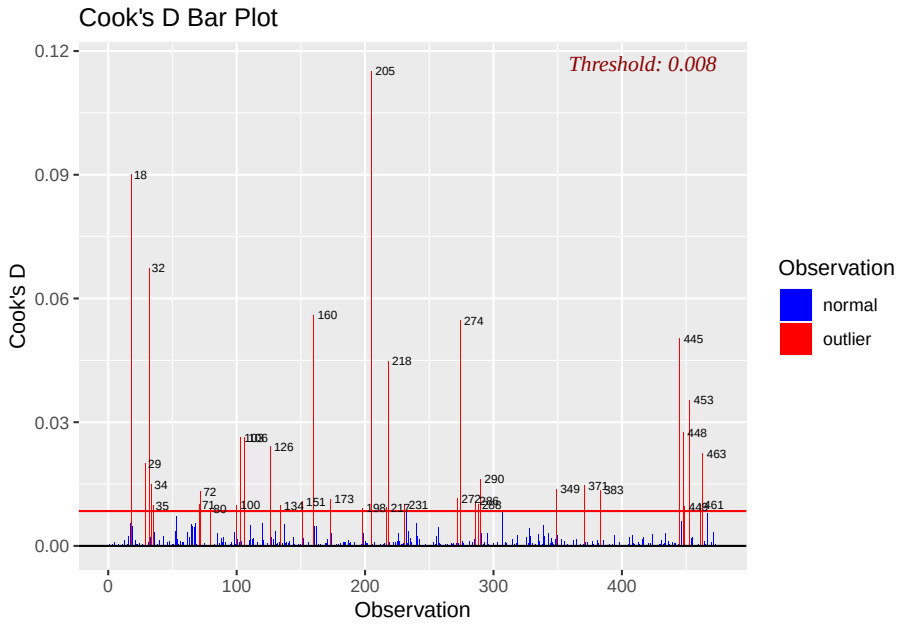
```
ols_plot_resid_qq(wybrany_model)
```



Gdyby wszystkie punkty leżały na prostej to oznaczałoby to normalność rozkładu reszt. Tymczasem po lewej i prawej stronie tego wykresu znajdują się potencjalne wartości odstające, które znacznie wpływają na rozkład reszt modelu.

Wartości odstające można ustalić na podstawie wielu kryteriów. Do jednych z najbardziej popularnych należy odległość Cooka:

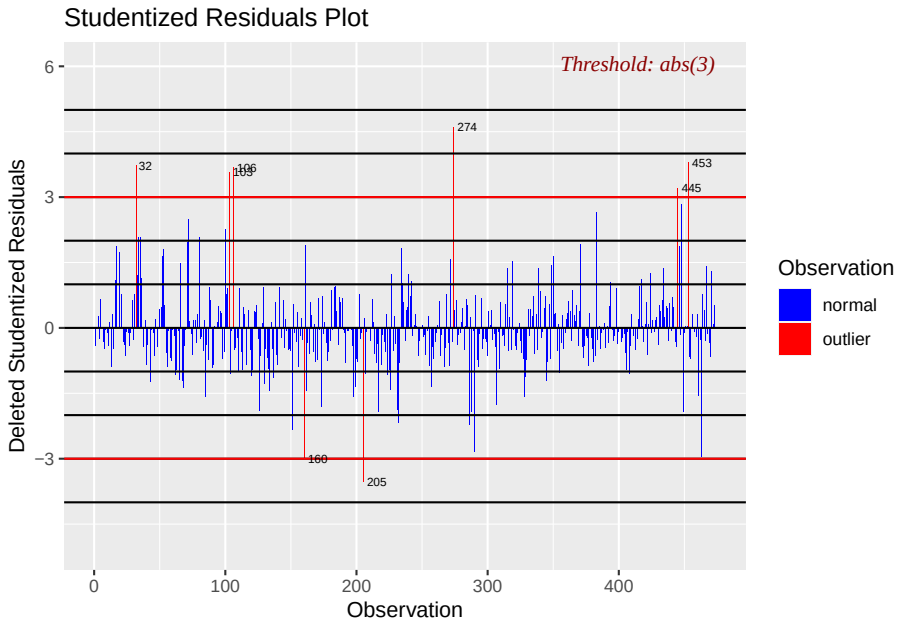
```
cook <- ols_plot_cooksd_bar(wybrany_model)
```



Przypisanie tej funkcji do obiektu zwraca nam tabelę z numerami zidentyfikowanych obserwacji wpływowych. W przypadku odległości Cooka jest to 35 obserwacji. Granica przynależności do grupy wartości odstających jest wyznaczana według wzoru: $4/n$, gdzie n to liczba wszystkich obserwacji.

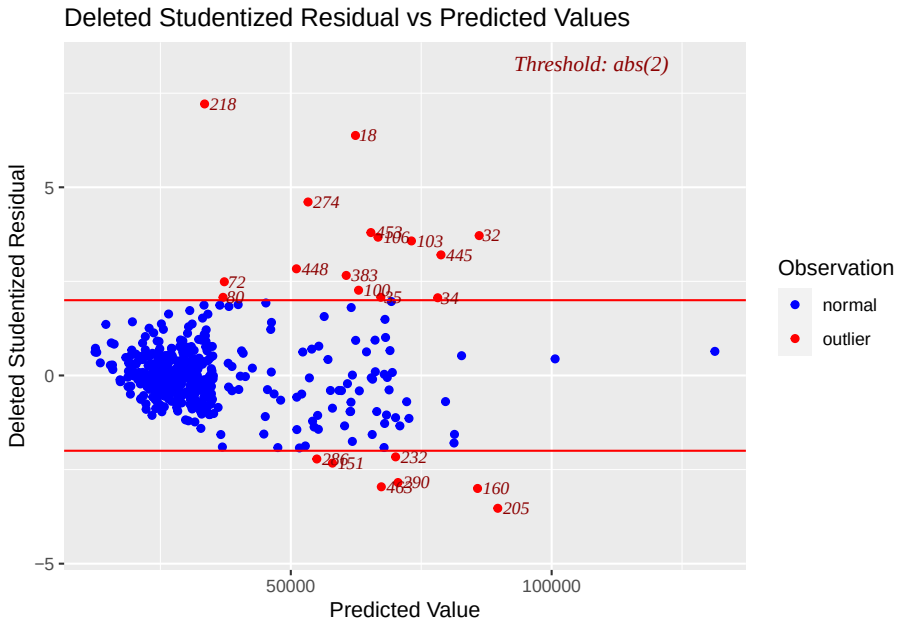
Inną miarą są reszty studentyzowane.

```
stud3 <- ols_plot_resid_stud(wybrany_model)
```



Wyżej wykorzystana funkcja jako kryterium odstawania przyjmuje wartość 3 identyfikując 10 obserwacji wpływowych. Z kolei dodanie do powyższej funkcji przyrostka *fit* powoduje przyjęcie jako granicy wartości równej 2.

```
stud2 <- ols_plot_resid_stud_fit(wybrany_model)
```



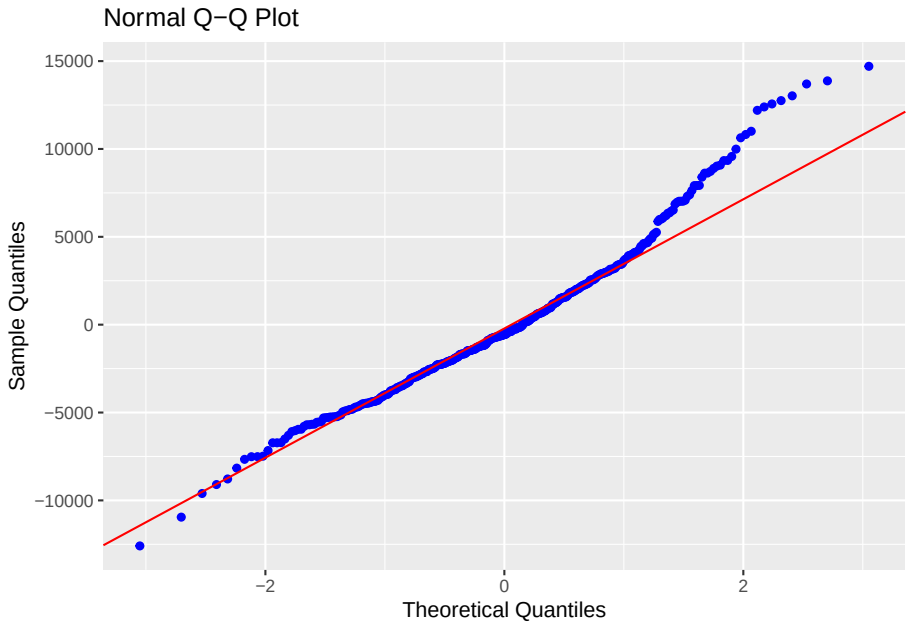
W ten sposób zostało zidentyfikowanych 22 obserwacji odstających. Korzystając z odległości Cooka wyeliminujemy obserwacje odstające ze zbioru:

```
outliers <- cook$data$obs[cook$data$color == "outlier"]

pracownicy_out <- pracownicy2[-outliers,]

wybrany_model_out <- lm(bwynagrodzenie ~ pwynagrodzenie + kat_pracownika + doswiadczeni
                        data = pracownicy_out)
summary(wybrany_model_out)
```

```
##
## Call:
## lm(formula = bwynagrodzenie ~ pwynagrodzenie + kat_pracownika +
##     doswiadczenie + staz + plec + edukacja, data = pracownicy_out)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12591.2  -2696.4   -564.8   2263.0  14704.1
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -3756.8288  2381.5361  -1.577 0.115420
## pwynagrodzenie      1.3696     0.0676  20.260  < 2e-16 ***
## kat_pracownika2  6480.6971  1049.9539   6.172 1.56e-09 ***
## kat_pracownika3  9791.5000  1059.4980   9.242  < 2e-16 ***
## doswiadczenie     -18.5808     2.3229  -7.999 1.17e-14 ***
## staz              116.3809    20.9704   5.550 5.01e-08 ***
## plec1           -1616.6030   505.1391  -3.200 0.001475 **
## edukacja          397.7652   105.8146   3.759 0.000194 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4271 on 430 degrees of freedom
## Multiple R-squared:  0.894,  Adjusted R-squared:  0.8923
## F-statistic: 518.3 on 7 and 430 DF,  p-value: < 2.2e-16
```

Model dopasowany na takim zbiorze charakteryzuje się dużo mniejszym błędem standardowym oraz wyższym współczynnikiem R^2 w porównaniu do poprzedniego. Sprawdźmy w takim razie normalność reszt.

```
ols_plot_resid_qq(wybrany_model_out)
```



Wykres kwantyl-kwantyl wygląda już dużo lepiej i w zasadzie nie obserwuje tu się dużych odstępstw od rozkładu normalnego. Przeprowadźmy jeszcze odpowiednie testy statystyczne.

```
ols_test_normality(wybrany_model_out)
```

```
## -----------------------------------------------
##        Test             Statistic       pvalue
## -----------------------------------------------
## Shapiro-Wilk              0.9696         0.0000
## Kolmogorov-Smirnov        0.0667         0.0405
## Cramer-von Mises         38.1644         0.0000
## Anderson-Darling          3.5421         0.0000
## -----------------------------------------------
```

Tylko test Kołmogorova-Smirnova na poziomie istotności $\alpha = 0,01$ wskazał na brak podstaw do odrzucenia hipotezy zerowej.

3.3.1 Zadanie

Na podstawie zbioru dotyczącego 50 startupów określ jakie czynniki wpływają na przychód startupów.

```
startupy <- read_xlsx("data/startups.xlsx")

startupy <- janitor::clean_names(startupy)
```

```
summary(startupy)
```

```
##    r_d_spend      administration   marketing_spend     state
##  Min.   :     0   Min.   : 51283   Min.   :     0   Length:50
##  1st Qu.: 39936   1st Qu.:103731   1st Qu.:129300   Class :character
##  Median : 73051   Median :122700   Median :212716   Mode  :character
##  Mean   : 73722   Mean   :121345   Mean   :211025
##  3rd Qu.:101603   3rd Qu.:144842   3rd Qu.:299469
##  Max.   :165349   Max.   :182646   Max.   :471784
##      profit
##  Min.   : 14681
##  1st Qu.: 90139
##  Median :107978
##  Mean   :112013
##  3rd Qu.:139766
##  Max.   :192262
```

Chapter 4

Grupowanie

Prezentacja

4.1 Wprowadzenie

Grupowanie polega na przypisanie obiektów do określonych grup/klastrów/skupień/segmentów, w których znajdą się jednostki najbardziej do siebie podobne, a powstałe grupy będą się między sobą różnić. Całe utrudnienie polega na tym, że nie wiemy ile tych grup ma powstać.

4.2 Metoda k-średnich

Najpopularniejszą metodą grupowania jest metoda k-średnich. Do jej zalet należy zaliczyć to, że dobrze działa zarówno na małych, jak i dużych zbiorach i jest bardzo efektywny - zwykle osiąga zbieżność w mniej niż 10 iteracjach. Z wad należy wskazać losowy wybór początkowych centrów skupień, co może skutkować nieprawidłowym przypisaniem obiektów do grup.

Algorytm postępowania jest następujący:

1. Wskaż liczbę grup k.
2. Wybierz dowolne k punktów jako centra grup.
3. Przypisz każdą z obserwacji do najbliższego centroidu.
4. Oblicz nowe centrum grupy.
5. Przypisz każdą z obserwacji do nowych centroidów. Jeśli któraś obserwacja zmieniła grupę - przejdź do kroku nr 4, a w przeciwnym przypadku zakończ algorytm.

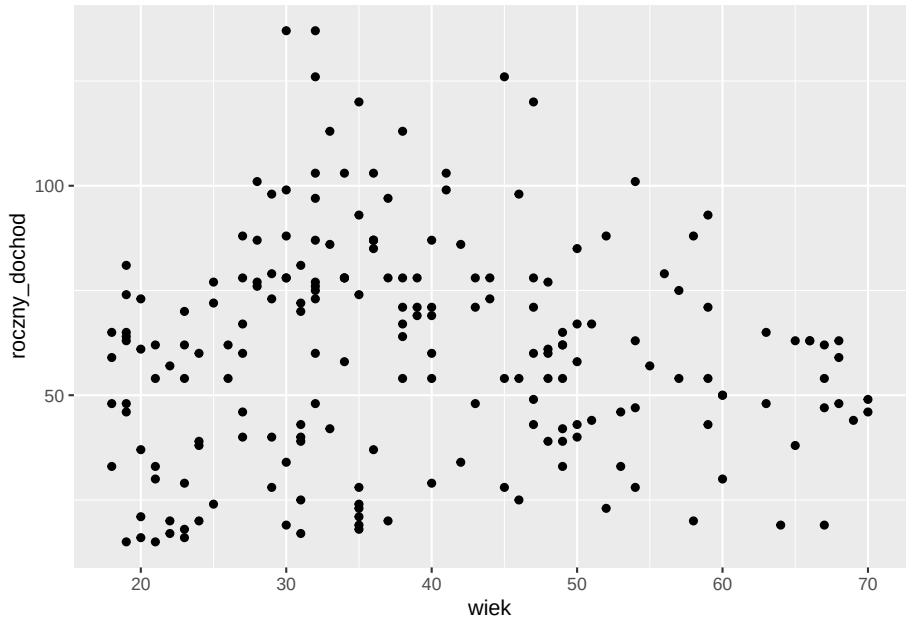
Wykorzystamy informacje ze zbioru zawierającego informacje o klientach sklepu i dokonamy grupowania tych klientów.

Opis zbioru:

- klientID - identyfikator klienta
- plec - płeć
- wiek - wiek
- roczny_dochod - roczny dochód wyrażony w tys. dolarów
- wskaznik_wydatkow - klasyfikacja sklepu od 1 do 100

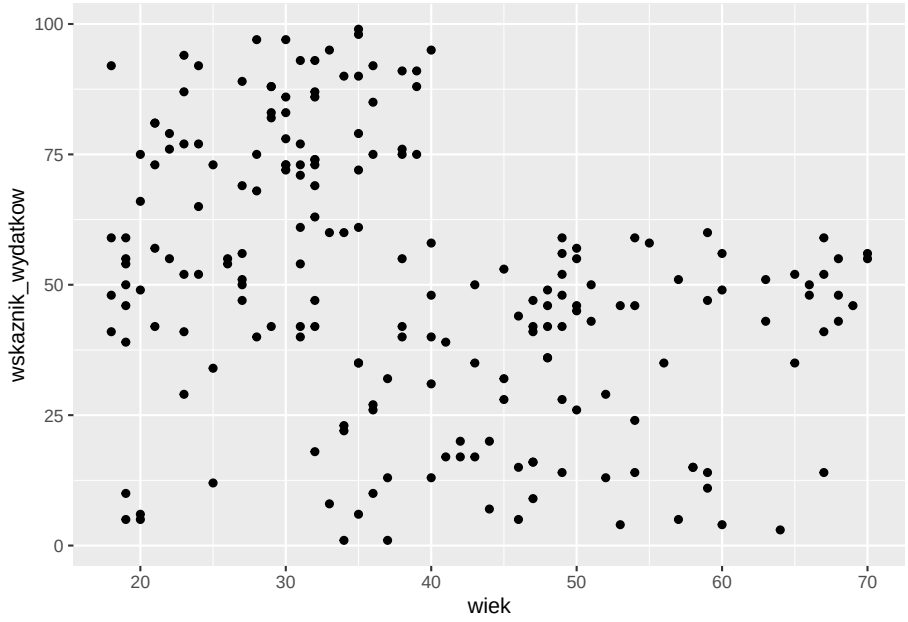
Wczytujemy zbiór danych i sprawdzamy czy pomiędzy zmiennymi są widoczne jakieś zależności.

```
library(tidyverse)

klienci <- read.csv("data/klienci.csv")

ggplot(klienci, aes(x=wiek, y=roczny_dochod)) +
  geom_point()
```



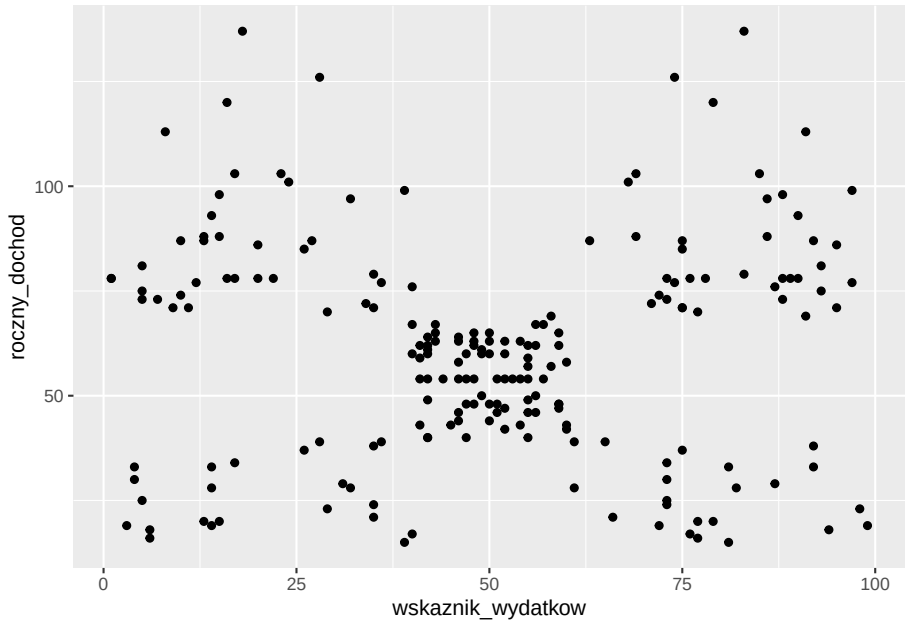
Pomiędzy wiekiem a rocznym dochodem nie widać zależności.

```
ggplot(klienci, aes(x=wiek, y=wskaznik_wydatkow)) +
  geom_point()
```



W przypadku wieku i wskaźnika wydatków moglibyśmy się pokusić o podział zbioru na dwie grupy za pomocą przekątnej.

```
ggplot(klienci, aes(x=wskaznik_wydatkow, y=roczny_dochod)) +
  geom_point()
```



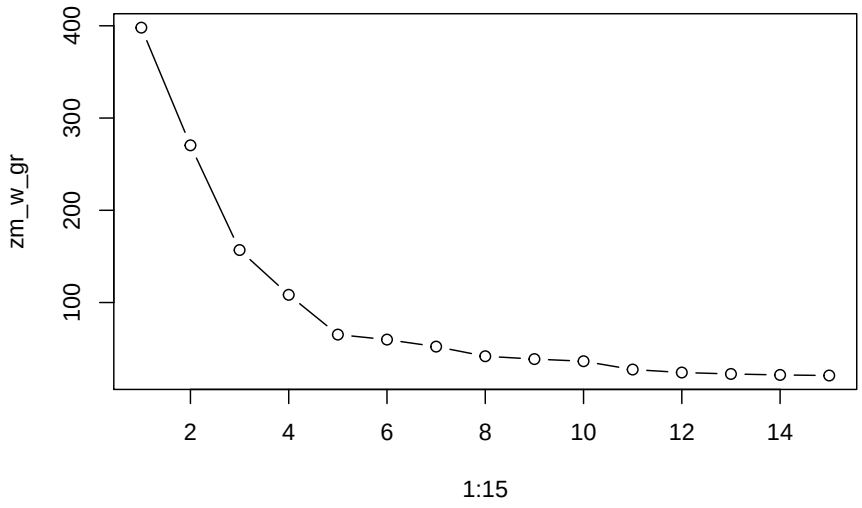
Po zestawieniu rocznego dochodu i wskaźnika wydatków wyłania się 5 potencjalnych grup, zatem te dwie cechy wykorzystamy do grupowania. Jednak przed zastosowaniem algorytmu musimy te dane przygotować normalizując zakres obu cech - w tym przypadku za pomocą standaryzacji.

```
klienci_z <- klienci %>%
  select(roczny_dochod, wskaznik_wydatkow) %>%
  scale()

head(klienci_z)
```

```
##      roczny_dochod wskaznik_wydatkow
## [1,]     -1.734646        -0.4337131
## [2,]     -1.734646         1.1927111
## [3,]     -1.696572        -1.7116178
## [4,]     -1.696572         1.0378135
## [5,]     -1.658498        -0.3949887
## [6,]     -1.658498         0.9990891
```

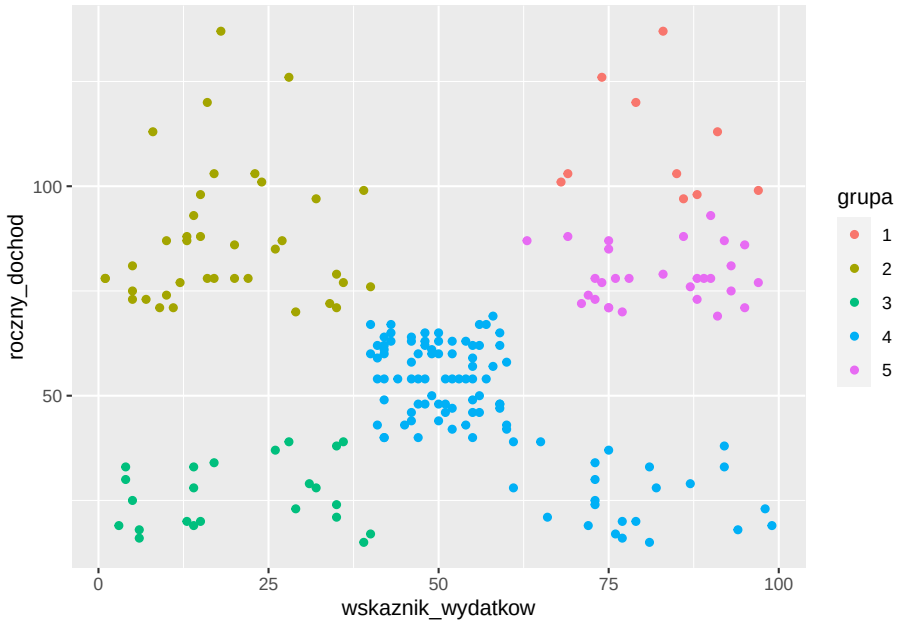
W przypadku, gdy podział na grupy nie jest tak oczywisty lub bierzemy pod uwagę więcej niż dwa kryteria to wówczas w wyznaczeniu optymalnej liczby skupień może pomóc wykres osypiska (ang. elbow method). Polega to na przeprowadzeniu grupowania (z wykorzystaniem funkcji `kmeans()`) dla różnej liczby grup i porównanie wariancji wewnątrz-grupowej. Dane do stworzenia wykresu osypiska możemy obliczyć w pętli:

```
zm_w_gr <- numeric(15)

# wprowadzenie pętli

for(i in 1:length(zm_w_gr)) {
  set.seed(14)
  gr <- kmeans(klienci_z, centers = i)
  zm_w_gr[i] <- gr$tot.withinss
}

plot(1:15, zm_w_gr, type="b")
```



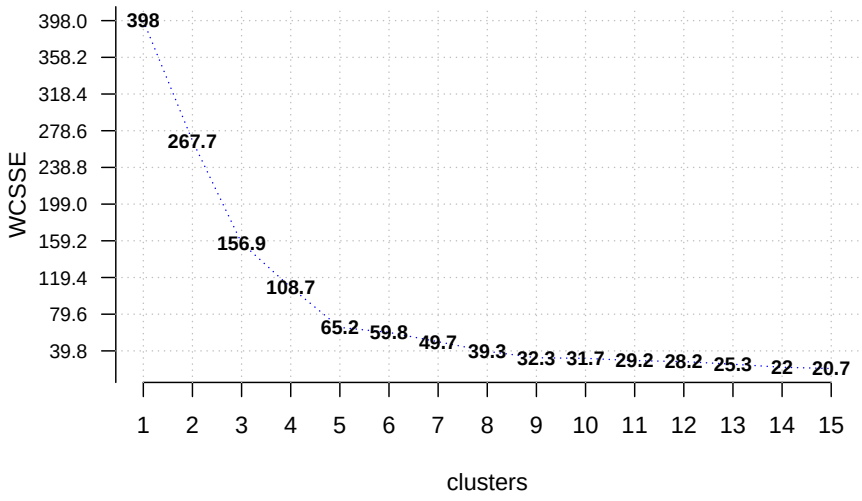
Wybieramy liczbę skupień po której nie następuje już gwałtowny spadek wartości wariancji wewnątrz-grupowej. Według tego kryterium powinniśmy wybrać wartość 6 zamiast 5. Sprawdźmy zatem jakie otrzymamy przyporządkowanie do grup. Następnie informację o tym przypisaniu umieszczamy w oryginalnym zbiorze danych i przedstawiamy na wykresie. W celu uzyskania powtarzalnych wyników zastosujemy stałe ziarno generatora.

```
set.seed(12)
grupy <- kmeans(x = klienci_z, centers = 5)

klienci$grupa <- as.factor(grupy$cluster)

ggplot(klienci, aes(x=wskaznik_wydatkow,
                    y=roczny_dochod,
                    color=grupa)) +
  geom_point()
```



Jak zauważamy ten podział nie jest właściwy. Ze względu na losowy przydział centrów skupień na początku algorytmu istnieje spora szansa, że rozwiązanie nie będzie optymalne. Rozwiązaniem tego problemu jest użycie algorytmu kmeans++ do początkowego ustalenia centrów. Ta metoda jest zaimplementowana w pakiecie `ClusterR`. Ponadto jest tam także funkcja do ustalenia optymalnej liczby skupień na podstawie wykresu osypiska.

```
library(ClusterR)

Optimal_Clusters_KMeans(data = klienci_z, max_clusters = 15, criterion = "WCSSE")
```

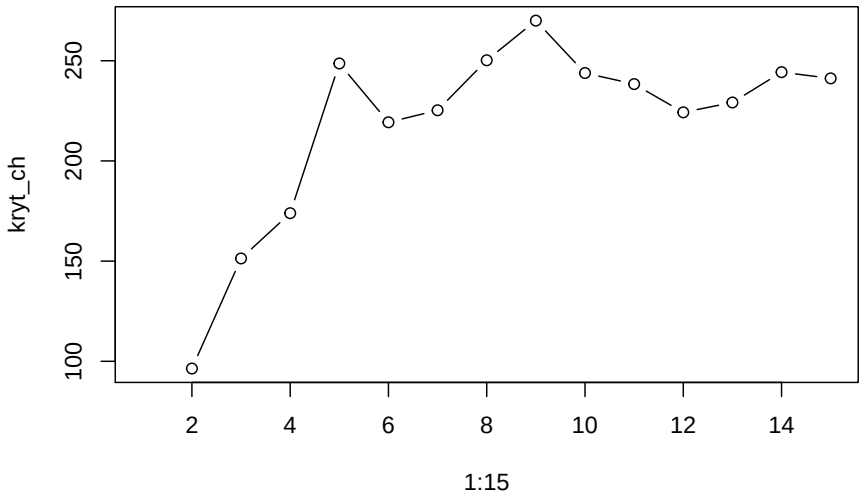



```
##  [1] 398.00000 267.67171 156.91549 108.68209  65.24057  59.83221  49.72816
##  [8]  39.31585  32.33261  31.70877  29.24003  28.18491  25.34096  22.01873
## [15]  20.67234
## attr(,"class")
## [1] "k-means clustering"
```

Wybieramy liczbę skupień po której nie następuje już gwałtowny spadek wartości wariancji wewnątrz-grupowej. W analizowanym przypadku będzie to 5 grup.

Dodatkowo można obliczyć jedno z wielu kryteriów dobroci podziału obiektów na grupy. Jednym z najpopularniejszych jest kryterium Calińskiego-Harabasza zaimplementowane m.in. w pakiecie clusterCrit. Podobnie jak w przypadku wykresu osypiska należy policzyć wartość tego kryterium dla różnej liczby segemntów i wybrać liczbę grup wskazaną przez wartość maksymalną.

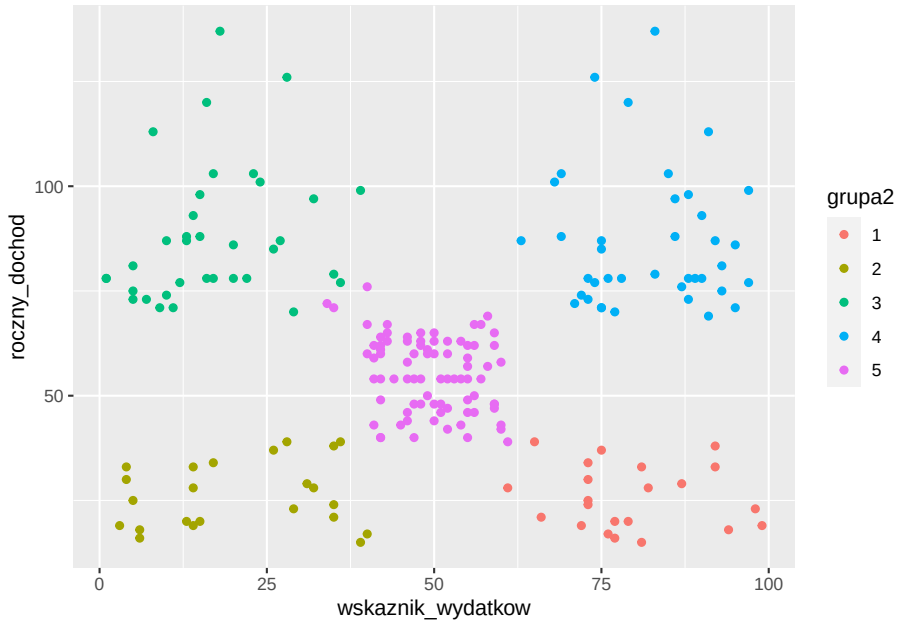
```
library(clusterCrit)

kryt_ch <- numeric(15)

for(i in 1:length(kryt_ch)) {
  gr <- KMeans_rcpp(klienci_z, clusters = i)
  kryt_ch[i] <- intCriteria(traj = klienci_z, part = as.integer(gr$clusters), crit = "cal")
}

plot(1:15, kryt_ch, type="b")
```



Najwyższe wartości indeksu Calińskiego-Harabasza obserwujemy dla 5 i 9 skupień. Należy jednak pamiętać, że w grupowaniu bardzo ważna jest ocena badacza pod kątem sensowności otrzymanego podziału - łatwiej stworzyć 5 różnych kampanii reklamowych aniżeli 9.

Następnie korzystamy z funkcji `KMeans_rcpp` do wyznaczenia przynależności do grup. Ta funkcja domyślnie korzysta z algorytmu kmeans++, zatem nie ma niebezpieczeństwa, że uzyskamy niewłaściwe przyporządkowanie.

```
grupy2 <- KMeans_rcpp(data = klienci_z, clusters = 5)

klienci$grupa2 <- as.factor(grupy2$clusters)

ggplot(klienci, aes(x=wskaznik_wydatkow,
                    y=roczny_dochod,
                    color=grupa2)) +
  geom_point()
```



Ostatnim etapem analizy jest odpowiednia charakterystyka uzyskanych klastrów - najczęściej wyznacza się średnie wartości cech w ramach każdej grupy.

```
klienci %>%
  select(-klientID, -plec, -grupa) %>%
  group_by(grupa2) %>%
  summarise_all(.funs = "mean")
```

```
## # A tibble: 5 x 4
##   grupa2  wiek roczny_dochod wskaznik_wydatkow
##   <fct>  <dbl>         <dbl>              <dbl>
## 1 1       25.3          25.7               79.4
## 2 2       45.2          26.3               20.9
## 3 3       41.1          88.2               17.1
## 4 4       32.7          86.5               82.1
## 5 5       42.7          55.3               49.5
```

W grupie pierwszej znalazły się osoby z niskimi dochodami i wysokim wskaźnikiem wydatków. Grupa druga to klienci o niskich dochodach i wydatkach - ich przeciwieństwem jest grupa 4. W grupie 3 są osoby z wysokimi dochodami, ale niskimi wydatkami. Grupa 5 to z kolei średniacy - klienci o średnich dochodach i wydatkach.

4.3 Metoda hierarchiczna

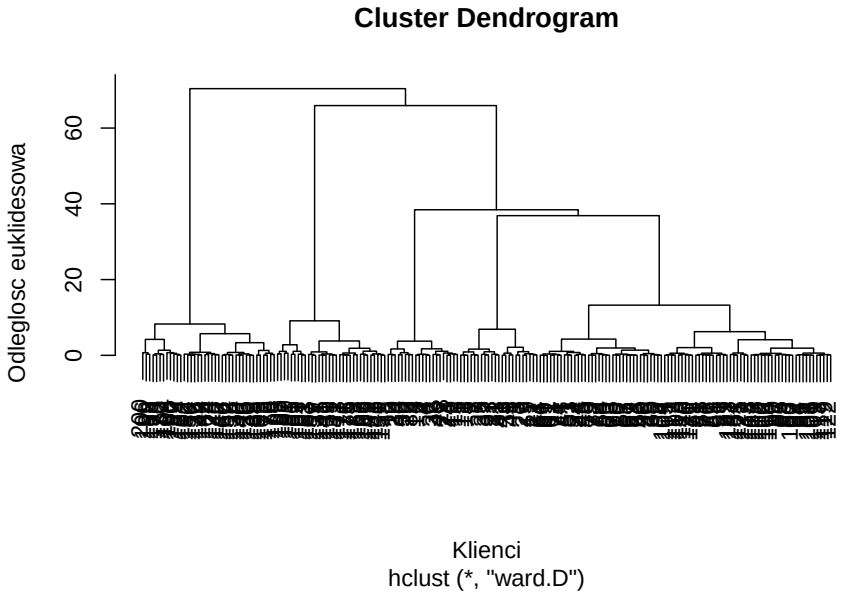
Alternatywną metodą grupowania jest metoda hierarchiczna. Do jej zalet zaliczymy prosty sposób ustalenia liczby grup oraz praktyczny sposób wizualizacji. Niestety nie jest to metoda odpowiednia dla dużych zbiorów danych.

Algorytm postępowania:

1. Każda obserwacji stanowi jedną z N pojedynczych grup.
2. Na podstawie macierzy odległości połącz dwie najbliżej leżące obserwacje w jedną grupę ($N-1$ grup).
3. Połącz dwa najbliżej siebie leżące grupy w jedną ($N-2$ grup).
4. Powtórz krok nr 3, aż do uzyskania jednej grupy.

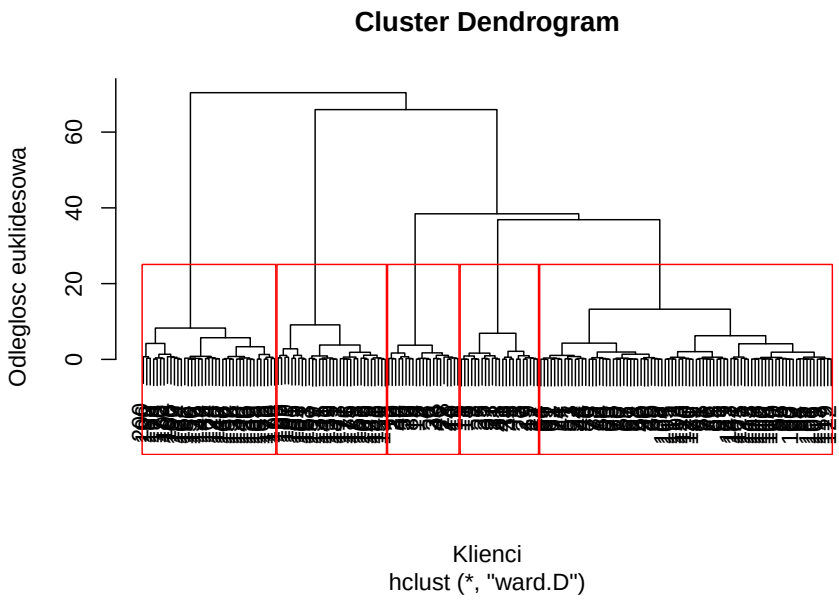
Dla tych samych danych przeprowadzimy grupowanie, ale tym razem metodą hierarchiczną. W metodzie hierarchicznej bazuje się na macierzy odległości pomiędzy obserwacjami. Można zastosować wiele miar odległości, ale najczęściej wykorzystywana jest odległość euklidesowa. Druga zmienna, na którą mamy wpływ to metoda łączenia skupień - w tym przypadku najlepsze rezultaty daje metoda Warda. Z kolei wyniki grupowania metodą hierarchiczną są prezentowane na dendrogramie.

```
macierz_odl <- dist(klienci_z)

dendrogram <- hclust(macierz_odl, method = "ward.D")

plot(dendrogram, xlab="Klienci", ylab="Odległość euklidesowa")
```



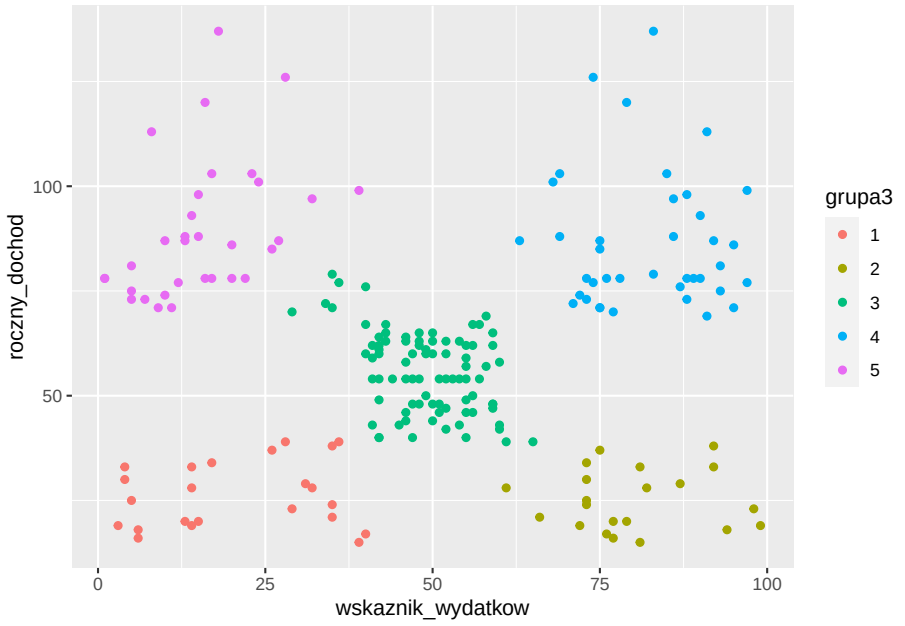
Na podstawie dendrogramu identyfikujemy największe różnice odległości opisane na osi Y. Także w tym przypadku identyfikujemy 5 grup. Istnieje także wiele kryteriów, które mają na celu wyznaczyć optymalną liczbę grup - link.

```
plot(dendrogram, xlab="Klienci", ylab="Odległość euklidesowa")
rect.hclust(dendrogram, k=5, border="red")
```



Następnie dopisujemy do oryginalnego zbioru danych etykiety utworzonych grup.

```
grupy_dendro <- cutree(dendrogram, 5)

klienci$grupa3 <- as.factor(grupy_dendro)

ggplot(klienci, aes(x=wskaznik_wydatkow,
                    y=roczny_dochod,
                    color=grupa3)) +
  geom_point()
```



Uzyskane wyniki są bardzo zbliżone do tych otrzymanych za pomocą algorytmu k-średnich.

```
klienci %>%
  select(-klientID, -plec, -grupa, -grupa2) %>%
  group_by(grupa3) %>%
  summarise_all(.funs = "mean")
```

```
## # A tibble: 5 x 4
##   grupa3  wiek roczny_dochod wskaznik_wydatkow
##   <fct>  <dbl>         <dbl>              <dbl>
## 1 1       45.2          26.3               20.9
## 2 2       25.3          25.1               80.0
## 3 3       42.5          55.8               49.1
## 4 4       32.7          86.5               82.1
## 5 5       41            89.4               15.6
```

Metoda hierarchiczna zastosowała inną numerację grup. Liczebności tych grup nieznacznie się różnią, ale charakterystyki wewnątrz grupowe są bardzo podobne do tych określonych na podstawie metody k-średnich.

Tworząc tabelę krzyżową możemy zobaczyć, że tylko 4 obserwacje zmieniły przypisanie do grup.

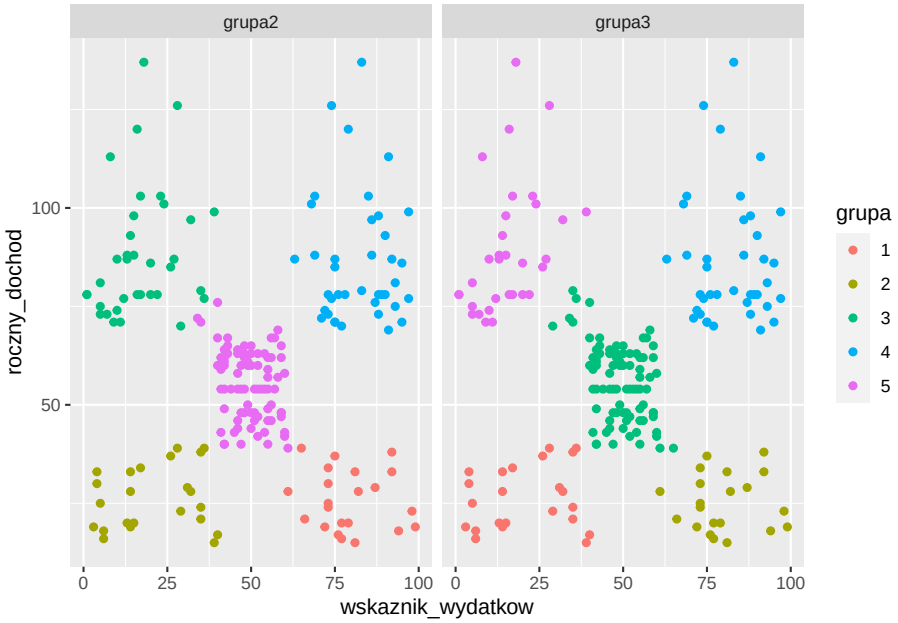
```
table(klienci$grupa2, klienci$grupa3)
```

```
##
```

```
##       1  2  3  4  5
##   1   0 21  1  0  0
##   2  23  0  0  0  0
##   3   0  0  3  0 32
##   4   0  0  0 39  0
##   5   0  0 81  0  0
```

Porównajmy jeszcze wyniki działania tych dwóch metod na wykresach:

```
klienci %>%
  select(roczny_dochod, wskaznik_wydatkow, grupa2, grupa3) %>%
  gather(metoda, grupa, -roczny_dochod, -wskaznik_wydatkow) %>%
  ggplot(aes(x=wskaznik_wydatkow, y=roczny_dochod)) +
  geom_point(aes(color=grupa)) +
  facet_wrap(~ metoda)
```



Problematyczne obserwacje pochodziły z grupy klientów o przeciętnych dochodach oraz wydatkach.

4.3.1 Zadania

1. Dokonaj grupowania danych dotyczących 32 samochodów według następujących zmiennych: pojemność, przebieg, lata oraz cena.
2. Rozpoznawanie czynności na podstawie danych z przyspieszeniomierza w telefonie: User Identification From Walking Activity Data Set

Chapter 5

Klasyfikacja

Prezentacja

5.1 Wprowadzenie

Celem analizy klasyfikacji jest zbudowanie modelu predykcyjnego, który w rezultacie zwróci prawdopodobieństwo przynależności danej obserwacji do jeden z dwóch klas. Przykładowo na podstawie danych o klientach banku można stworzyć model oceny zdolności kredytowej dla nowych klientów. W tym rozdziale posłużymy się German Credit Data do budowy takiego predyktora. Wybrane kolumny z tego zbioru znajdują się w tym pliku.

W pierwszym kroku wczytujemy dane i dokonujemy niezbędnych przekształceń tego zbioru. Z wykorzystaniem funkcji `clean_names()` z pakietu *janitor* zamieniamy nazwy kolumn w przyjazne przetwarzaniu przez komputer (brak spacji, polskich znaków, itp.). Następnie zmienne tekstowe zamieniamy na zmienne jakościowe - faktory oraz tworzymy nową kolumnę zawierającą wysokość raty kredytu.

```
library(tidyverse)
library(readxl)
library(janitor)

credit <- read_xlsx("data/german_credit_data.xlsx") %>%
  clean_names() %>%
  mutate_if(is.character, as.factor) %>%
  mutate(job=as.factor(job),
         installment=credit_amount/duration)

summary(credit)
```

```
##       age            sex      job     housing      saving_accounts
##  Min.   :19.00   female:310   0: 22   free:108   little    :603
##  1st Qu.:27.00   male  :690   1:200   own :713   moderate  :103
##  Median :33.00                2:630   rent:179   quite rich: 63
##  Mean   :35.55                3:148              rich      : 48
##  3rd Qu.:42.00                                   NA's      :183
##  Max.   :75.00
##
##  checking_account credit_amount      duration                  purpose
##  little  :274     Min.   :  250   Min.   : 4.0   car                :337
##  moderate:269     1st Qu.: 1366   1st Qu.:12.0   radio/TV           :280
##  rich    : 63     Median : 2320   Median :18.0   furniture/equipment:181
##  NA's    :394     Mean   : 3271   Mean   :20.9   business           : 97
##                   3rd Qu.: 3972   3rd Qu.:24.0   education          : 59
##                   Max.   :18424   Max.   :72.0   repairs            : 22
##                                                  (Other)            : 24
##    risk      installment
##  bad :300   Min.   :  24.06
##  good:700   1st Qu.:  89.60
##             Median : 130.33
##             Mean   : 167.69
##             3rd Qu.: 206.18
##             Max.   :2482.67
##
```

Zamiana cech tekstowych na faktory pozwala w podsumowaniu wygenerowanym przez funkcję `summary()` obserwować od razu częstości poszczególnych wariantów. Możemy zaobserwować występowanie braków danych w zmiennych saving accounts i checking_account. Generalnie jest to zbyt duży problem, bo tylko niektóre algorytmy klasyfikacji nie obsługują braków danych w zmiennych objaśniających. Ważne jest, żeby braki danych nie występowały dla cechy decyzyjnej.

Obserwacje z brakami danych można usunąć, ale często spowodowałoby to znaczne zmniejszenie próby badawczej, zatem stosuje się metody mające na celu uzupełnienie braków danych. W najprostszym przypadku braki można zastąpić średnią, medianą lub dominantą. Do bardziej zaawansowanych sposobów należy metoda najbliższych sąsiadów (VIM) albo imputacja wielokrotna (mice).

W omawianym przypadku klientów wiarygodnych jest 700, a tych, którzy nie spłacili zobowiązania 300. Mamy zatem do czynienia z niezbalansowaną próbą. W idealnym przypadku klasyfikacji, przypadków z każdej grupy powinno być tyle samo. W przeciwnym przypadku model będzie działał lepiej dla klasy większościowej. Najprostszą metodą balansowania danych jest upsampling czyli dolosowywanie obserwacji z klasy mniejszościowej, tak aby wyrównać liczebności. Przeciwieństwem tego podejścia jest downsampling. Alternatywnie można zastosować metodę SMOTE, która generuje sztuczne obserwacje dla klasy mniejs-

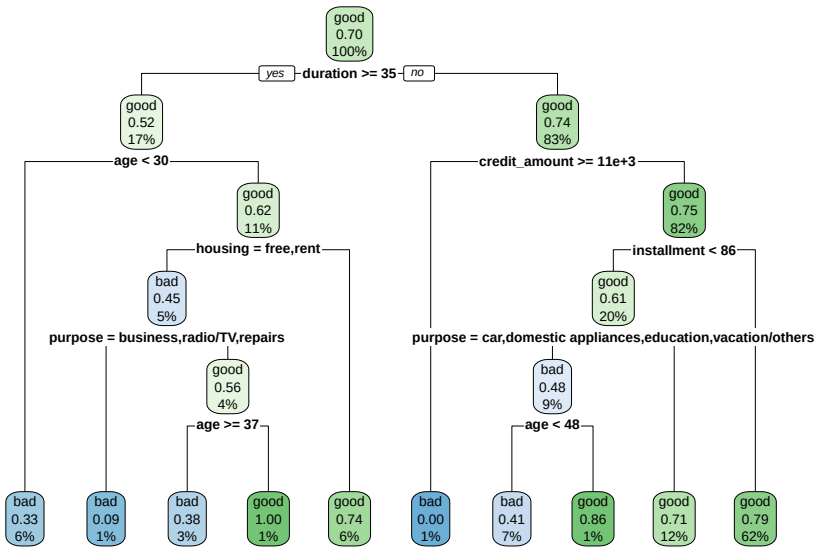
zościowej (pakiety DMwR, imbalance).

5.2 Drzewa decyzyjne

Najpopularniejszą metodą klasyfikacji są drzewa decyzyjne, które charakteryzują się z reguły dobrą efektywnością i pozwalają na łatwą interpretację zastosowanych reguł klasyfikacji. Wykorzystamy pakiet rpart do stworzenia drzewa oraz pakiet rpart.plot do wizualizacji.

Proces tworzenia drzewa jest bardzo prosty, a argumenty w funkcji `rpart()` są takie same jak w regresji liniowej.

```
library(rpart)
library(rpart.plot)

m1 <- rpart(formula = risk ~ ., data = credit)

rpart.plot(m1)
```



W rezultacie uzyskujemy drzewo decyzyjne z optymalnie dobranymi zmiennymi objaśniającymi. W każdym węźle drzewa podane są następujące wartości: - prognozowana klasa, prawdopodobieństwa zaklasyfikowania do klasy pozytywnej, odsetek obserwacji w węźle. Domyślnym progiem klasyfikacji jest wartość 0,5. Jeżeli prawdopodobieństwo jest poniżej tej wartości to nastąpi przypisanie do grupy klientów niespłacających pożyczki, a jeśli powyżej to do tej drugiej grupy.

Oceny jakości klasyfikatora dokonuje się na podstawie macierzy pomyłek oraz miar wyznaczonych na jej podstawie. Najpopularniejsze z nich to:

- dokładność (accuracy): % poprawnie zaklasyfikowanych
- precyzja (precison): % poprawnie rozpoznanych przypadków pozytywnych TP/(TP+FP)
- czułość (sensitivity/recall): % prawdziwie pozytywnych TP/(TP+FN)
- swoistość (specificity): % prawdziwie negatywnych TN/(TN+FP)
- F1: średnia harmoniczna z czułości i precyzji 2TP/(2TP+FP+FN)

Im wyższe wartości tych miar tym lepszy klasyfikator. Do wyznaczenia tych miar w R służy funkcja z pakietu caret. W tym celu trzeba wyznaczyć wartości prognozowanej klasy na podstawie modelu.

```
library(caret)

pred_risk_m1 <- predict(object = m1, newdata = credit, type = "class")
```

Argument `type` określa typ predykcji: `"class"` oznacza prognozowaną klasę, a `"prob"` prawdopodobieństwo. Na tej podstawie oraz wartości rzeczywistych tworzymy macierz pomyłek:

```
confusionMatrix(data = pred_risk_m1, reference = credit$risk, positive = "good", mode =
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad  117   60
##       good 183  640
##
##                Accuracy : 0.757
##                  95% CI : (0.7292, 0.7833)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 3.553e-05
##
##                   Kappa : 0.3447
##
##  Mcnemar's Test P-Value : 5.024e-15
##
##             Sensitivity : 0.9143
##             Specificity : 0.3900
##          Pos Pred Value : 0.7776
##          Neg Pred Value : 0.6610
##               Precision : 0.7776
##                  Recall : 0.9143
##                      F1 : 0.8404
##              Prevalence : 0.7000
```

```
##          Detection Rate : 0.6400
##    Detection Prevalence : 0.8230
##       Balanced Accuracy : 0.6521
##
##        'Positive' Class : good
##
```

W naszym przykładzie spośród 1000 klientów, model jako wiarygodnych kredytobiorców zaklasyfikował 640, a 117 prawidłowo jako osoby, które nie spłaciły zobowiązania. W 60 przypadkach model uznał brak zdolności kredytowej u klienta, podczas gdy w rzeczywistości pożyczka została spłacona. Dla 183 klientów podjętoby odwrotną decyzję - model przyznałby kredyt, a w rzeczywistości osoby te nie spłaciły pożyczki. Dokładność w tym przypadku wynosi 75,7%, a precyzja 77,8%. Czułość tego predyktora jest wysoka (91,4%), ale swoistość już nie (39%), na co może mieć wpływ niezbalansowanie danych.

Przedstawiony powyżej przykład miał charakter analizy ekspolarycyjnej - opartej na całym zbiorze danych. W praktyce stosuje się podejście polegające na podziale zbioru danych na zbiór treningowy oraz walidacyjny. Na danych ze zbioru treningowego buduje się model, który następnie testowany jest na danych, których nigdy wcześniej "nie widział" - na zbiorze walidacyjnym. Miary klasyfikacji obliczone na podstawie zbioru walidacyjnego dostarczają realnej oceny jakości klasyfikatora.

Do podziału zbioru służy funkcja `createDataPartition()` z pakietu *caret*. Zbiory treningowy i walidacyjny tworzone są w taki sposób, aby zachować proporcje w zmiennej decyzyjnej. Domyślnie funkcja dzieli zbiór danych w układzie 50/50, natomiast w tym przykładzie 80% obserwacji umieścimy w zbiorze treningowym.

```
set.seed(123)
split <- createDataPartition(y = credit$risk, p = 0.8)

train_credit <- credit[split$Resample1,]
valid_credit <- credit[-split$Resample1,]

summary(train_credit)
```

```
##       age            sex      job     housing      saving_accounts
##  Min.   :19.00   female:252   0: 17   free: 80   little    :486
##  1st Qu.:27.00   male  :548   1:159   own :576   moderate  : 85
##  Median :33.00                2:504   rent:144   quite rich: 49
##  Mean   :35.43                3:120              rich      : 34
##  3rd Qu.:42.00                                   NA's      :146
##  Max.   :75.00
##
##  checking_account credit_amount      duration                     purpose
##  little  :228     Min.   :  276   Min.   : 4.00   car                :284
```

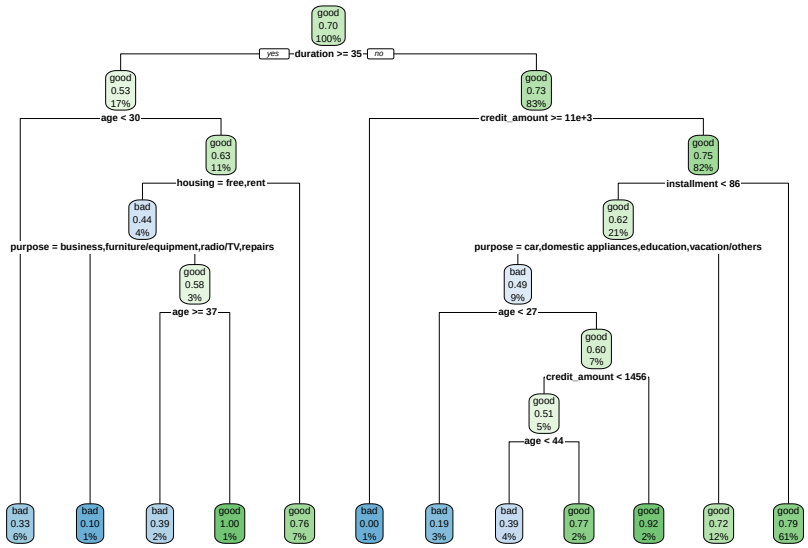
```
##  moderate:205     1st Qu.: 1374   1st Qu.:12.00   radio/TV           :219
##  rich    : 46     Median : 2326   Median :18.00   furniture/equipment:139
##  NA's    :321     Mean   : 3300   Mean   :21.02   business           : 76
##                   3rd Qu.: 3965   3rd Qu.:24.00   education          : 46
##                   Max.   :18424   Max.   :72.00   repairs            : 17
##                                                   (Other)            : 19
##    risk      installment
##  bad :240   Min.   :  24.06
##  good:560   1st Qu.:  87.18
##             Median : 131.74
##             Mean   : 168.95
##             3rd Qu.: 206.06
##             Max.   :2482.67
##
```

```
summary(valid_credit)
```

```
##       age            sex      job     housing      saving_accounts
##  Min.   :20.00   female: 58   0:  5   free: 28   little    :117
##  1st Qu.:27.00   male  :142   1: 41   own :137   moderate  : 18
##  Median :34.00                2:126   rent: 35   quite rich: 14
##  Mean   :36.02                3: 28              rich      : 14
##  3rd Qu.:42.00                                   NA's      : 37
##  Max.   :74.00
##
##  checking_account credit_amount      duration                     purpose
##  little  :46      Min.   :  250   Min.   : 4.00   radio/TV           :61
##  moderate:64      1st Qu.: 1308   1st Qu.:12.00   car                :53
##  rich    :17      Median : 2261   Median :18.00   furniture/equipment:42
##  NA's    :73      Mean   : 3157   Mean   :20.43   business           :21
##                   3rd Qu.: 4003   3rd Qu.:24.00   education          :13
##                   Max.   :15672   Max.   :48.00   repairs            : 5
##                                                   (Other)            : 5
##    risk      installment
##  bad : 60   Min.   : 38.12
##  good:140   1st Qu.: 93.12
##             Median :127.23
##             Mean   :162.62
##             3rd Qu.:206.21
##             Max.   :730.80
##
```

Wykorzystując tak przygotowane dane możemy jeszcze raz wykorzystać drzewa decyzyjne do stworzenia klasyfikatora, ale tym razem wyłącznie na zbiorze treningowym.

```
m2 <- rpart(risk ~ ., train_credit)

rpart.plot(m2)
```



Utworzone drzewo będzie różnić się od tego, które powstało na podstawie całego zbioru danych. Następnie obliczamy prognozowane klasy na obu zbiorach i wyznaczamy macierze pomyłek.

```
pred_risk_m2_train <- predict(object = m2, newdata = train_credit, type = "class")
pred_risk_m2_valid <- predict(object = m2, newdata = valid_credit, type = "class")

# train
confusionMatrix(data = pred_risk_m2_train, reference = train_credit$risk,
                positive = "good", mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad   94   38
##       good 146  522
##
##                Accuracy : 0.77
##                  95% CI : (0.7392, 0.7987)
##     No Information Rate : 0.7
```

```
##     P-Value [Acc > NIR] : 5.783e-06
##
##                   Kappa : 0.3716
##
##  Mcnemar's Test P-Value : 3.067e-15
##
##             Sensitivity : 0.9321
##             Specificity : 0.3917
##          Pos Pred Value : 0.7814
##          Neg Pred Value : 0.7121
##               Precision : 0.7814
##                  Recall : 0.9321
##                      F1 : 0.8502
##              Prevalence : 0.7000
##          Detection Rate : 0.6525
##    Detection Prevalence : 0.8350
##       Balanced Accuracy : 0.6619
##
##        'Positive' Class : good
##
```

```
# valid
confusionMatrix(data = pred_risk_m2_valid, reference = valid_credit$risk,
                positive = "good", mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction bad good
##       bad   18   11
##       good  42  129
##
##                Accuracy : 0.735
##                  95% CI : (0.6681, 0.7948)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.1579
##
##                   Kappa : 0.2598
##
##  Mcnemar's Test P-Value : 0.00003775
##
##             Sensitivity : 0.9214
##             Specificity : 0.3000
##          Pos Pred Value : 0.7544
##          Neg Pred Value : 0.6207
##               Precision : 0.7544
```

```
##                  Recall : 0.9214
##                      F1 : 0.8296
##              Prevalence : 0.7000
##          Detection Rate : 0.6450
##    Detection Prevalence : 0.8550
##       Balanced Accuracy : 0.6107
##
##        'Positive' Class : good
##
```

Generalnie wyniki w obu przypadkach powinny być do siebie zbliżone, przy czym na zbiorze walidacyjnym miary jakości predycji mogą być trochę gorsze. Bardzo wysoka wartość dokładność na zbiorze treningowym, a niska na zbiorze walidacyjnym jest symptomem przeuczenia modelu - algorytm nauczył się odpowiedzi "na pamięć".

5.3 Gradient Boosting Machine

Spróbujemy polepszyć jakość klasyfikacji z wykorzystaniem metody gradient boostingu. W tym celu wykorzystamy pakiet h2o, który dostarcza kompleksowych rozwiązań z zakresu machine learning. W pierwszej kolejności trzeba zmienić format danych na ten obsługowany przez pakiet.

```
library(h2o)

h2o.init()
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         7 minutes 37 seconds
##     H2O cluster timezone:       Europe/Warsaw
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.30.0.1
##     H2O cluster version age:    1 month and 29 days
##     H2O cluster name:           H2O_started_from_R_lukas_xpm862
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   3.98 GB
##     H2O cluster total cores:    8
##     H2O cluster allowed cores:  8
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     H2O API Extensions:         Amazon S3, Algos, AutoML, Core V3, TargetEncoder, Core V4
```

```
##     R Version:                  R version 4.0.0 (2020-04-24)
```

```
h2o.no_progress() # brak pasków postępu

train_credit_h2o <- as.h2o(train_credit)
valid_credit_h2o <- as.h2o(valid_credit)
```

Następnie deklarujemy nazwy wykorzystywanych zmiennych i uruchamiamy procedurę:

```
y_var <- "risk"
x_var <- names(credit)[-10]

m3 <- h2o.gbm(x = x_var,
              y = y_var,
              training_frame = train_credit_h2o,
              validation_frame = valid_credit_h2o,
              seed = 1)

m3
```

```
## Model Details:
## ==============
##
## H2OBinomialModel: gbm
## Model ID:  GBM_model_R_1591105482481_310
## Model Summary:
##   number_of_trees number_of_internal_trees model_size_in_bytes min_depth
## 1              50                        50               13914         5
##   max_depth mean_depth min_leaves max_leaves mean_leaves
## 1         5    5.00000          7         26    17.66000
##
##
## H2OBinomialMetrics: gbm
## ** Reported on training data. **
##
## MSE:  0.07736868
## RMSE:  0.2781523
## LogLoss:  0.2767961
## Mean Per-Class Error:  0.09880952
## AUC:  0.9704167
## AUCPR:  0.9862121
## Gini:  0.9408333
## R^2:  0.6315777
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal threshold:
##        bad good    Error     Rate
```

```
## bad    202   38 0.158333  =38/240
## good    22  538 0.039286  =22/560
## Totals 224  576 0.075000  =60/800
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold      value idx
## 1                        max f1  0.568378   0.947183 232
## 2                        max f2  0.425774   0.965217 277
## 3                  max f0point5  0.642867   0.951409 205
## 4                  max accuracy  0.576117   0.925000 228
## 5                 max precision  0.985657   1.000000   0
## 6                    max recall  0.314379   1.000000 319
## 7               max specificity  0.985657   1.000000   0
## 8              max absolute_mcc  0.576117   0.819649 228
## 9   max min_per_class_accuracy  0.644367   0.912500 204
## 10 max mean_per_class_accuracy  0.642867   0.914286 205
## 11                      max tns  0.985657 240.000000   0
## 12                      max fns  0.985657 559.000000   0
## 13                      max fps  0.039037 240.000000 399
## 14                      max tps  0.314379 560.000000 319
## 15                      max tnr  0.985657   1.000000   0
## 16                      max fnr  0.985657   0.998214   0
## 17                      max fpr  0.039037   1.000000 399
## 18                      max tpr  0.314379   1.000000 319
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or `h2o.gainsLift(<model>, val
## H2OBinomialMetrics: gbm
## ** Reported on validation data. **
##
## MSE:  0.1742677
## RMSE:  0.4174538
## LogLoss:  0.5334558
## Mean Per-Class Error:  0.4166667
## AUC:  0.7427381
## AUCPR:  0.8549562
## Gini:  0.4854762
## R^2:  0.1701539
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal threshold:
##        bad good    Error     Rate
## bad     10   50 0.833333   =50/60
## good     0  140 0.000000   =0/140
## Totals  10  190 0.250000  =50/200
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold      value idx
```

```
## 1                       max f1  0.251100   0.848485 189
## 2                       max f2  0.251100   0.933333 189
## 3                 max f0point5  0.586949   0.823864 140
## 4                 max accuracy  0.540278   0.760000 149
## 5                max precision  0.977718   1.000000   0
## 6                   max recall  0.251100   1.000000 189
## 7              max specificity  0.977718   1.000000   0
## 8             max absolute_mcc  0.586949   0.413905 140
## 9   max min_per_class_accuracy  0.726448   0.700000 115
## 10 max mean_per_class_accuracy  0.586949   0.705952 140
## 11                      max tns  0.977718  60.000000   0
## 12                      max fns  0.977718 139.000000   0
## 13                      max fps  0.074617  60.000000 199
## 14                      max tps  0.251100 140.000000 189
## 15                      max tnr  0.977718   1.000000   0
## 16                      max fnr  0.977718   0.992857   0
## 17                      max fpr  0.074617   1.000000 199
## 18                      max tpr  0.251100   1.000000 189
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or `h2o.gainsLift(<m
```