Statystyczna Analiza Danych
zajęcia laboratoryjne 4

Wprowadzenie do grafiki

Bioinformatyka II rok

Wprowadzenie do grafiki w R – podstawowe funkcje

R ma szerokie możliwości w zakresie graficznego prezentowania danych. Podstawową funkcją do tworzenia wykresów jest funkcja plot(), która przyjmuje wiele argumentów pozwalających kontrolować wygląd wykresu, m.in.: typ wykresu, kolor, nazwy osi, tytuł wykresu. W pierwszej części pojawią się najczęściej używane argumenty funkcji plot(), dodatkowe funkcje przydatne przy tworzeniu wykresów: text(), abline(), points(), najczęściej używane parametry par(). Następnie pojawią się przykłady użycia różnych typów wykresów.

1 Funkcja plot() i jej podstawowe argumenty

Przykład 1.1 – funkcja plot():

Na początku przygotowujemy przykładowe dane, wykorzystujemy dostępne dane "cars" z pakietu datasets(). Na potrzeby wykresy wyselekcjonowano 10 pierwszych rekordów, a następnie wykorzystujemy funkcję plot() do narysowania podstawowego wykresu:

```
1 wybraneAuta <- head(cars, 10)
2 plot(wybraneAuta)
```

Przykład 1.2 – funkcja plot() z dodatkowymi argumentami:

Przykład 1.2 wykorzystuje funkcje plot() z dodatkowymi argumentami, aby poprawić wykres wizualnie. Dodano etykiety dla osi (xlab i ylab), dodano tytuł wykresu (main) oraz jego kolor (col.main), dodano podtytuł (sub) oraz kształt punktów (pch) i ich kolor (col). Spójrz jak szeroki zakres kolorów udostępnia R: colors() - http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf:

```
1 plot(x = wybraneAuta$speed, y = wybraneAuta$dist,
2      xlab = "szybkosc",
3      ylab = "dystans",
4      main = "Wykres szybkosc vs dystans",
5      col.main = "blue",
6      sub = "dane cars z pakietu datasets",
7      col = "blue", #zmiana koloru punktow, funkcja colors() pokazuje dostepne kolory
8      pch = 16 #ksztalt punktow, zobacz: ?pch
9 )
```

W przykładzie 1.2, wykorzystano tylko kilka z możliwych argumentów, można zmieniać nie tylko kolor tytułu (col.main), ale także kolor podtytułu (col.sub) czy etykiet osi (col.lab), do zapoznania się z innymi dodatkowymi argumentami odsyłam do R, sprawdź co ma do zaoferowania funkcja parametrów graficznych par().

W powyższym przykładzie użyliśmy ustalony kształtu symbolu (pch = 16). Niektóre z symboli mogą przyjmować dodatkowy kolor tła (argument bg), czyli wyróżniamy kolor obwódki i kolor wypełnienia, dla pch = 21:25. Spójrz jakie kształty przyjmują określone wartości: ?pch.

Przykład 1.3 – funkcja plot(), argument bg dla punktów na wykresie

Przykład 1.3 zawiera użycie kształtów z dodatkowym wypełnieniem:

```
plot(x = wybraneAuta$speed, y = wybraneAuta$dist,
     xlab = "szybkosc",
     ylab = "dystans",
     main = "Wykres szybkosc vs dystans",
     col.main = "blue",
     sub = "dane cars z pakietu datasets",
     col = "blue", #zmiana koloru punktow
     pch = 24, #ksztalt punktow, zobacz ?pch
     #pch = 21:25 umozliwiaja uzycie tla dla symbolu
     bg = "skyblue"
)
```

1.1 Typ wykresu - type

Do podstawowych elementów grafiki należy argument: type, który określa typ wykresu. Najbardziej popularne typy to: liniowy, punktowy, liniowy i punktowy jednocześnie. R umożliwia jednak użycie jeszcze innych typów wykresów, wykorzystaj poniższe przykłady aby przetestować możliwe warianty.

Przykład 1.4 – type

```
plot(wybraneAuta, type = "l")
```

Kolejnymi często używanymi argumentami są: lty i lwd, które pozwalają na dostosowanie linii na wykresie. Argument lty pozwala wybrać typy linii, podczas gdy argument lwd pozwala dostosować grubość linii. W przykładzie 1.5 wykorzystano typ linii 2, czyli przerywaną. Sprawdź jakie inne typy linii możemy wykorzystać w R.

Przykład 1.5 – lty i lwd

```
plot(wybraneAuta, type = "h", lwd = 10)
plot(wybraneAuta, type = "l", lty = 2)
```

1.2 Typ pudełka – box type

Obok wspomnianego już typu wykresu, należy wyróżnić jeszcze typ pudełka (box type). Typ pudełka (argument bty) jest kolejnym argumentem funkcji par(). Poniżej wybrane przykłady.

Przykład 1.6 – box type

```
plot(wybraneAuta, type = "b", bty = "o") #domyslne pudelko
plot(wybraneAuta, type = "b", bty = "n") #brak ksztaltu, same osie
```

1.3 Funkcja axis() - dostosowanie osi

Dostosowanie osi na wykresie można rozumieć na kilka sposobów. Jednym z nich dostosowanie skali osi x i y poprzez ustawienie wartości argumentów xlim i ylim. Innym sposobem jest dostosowanie typu linii osi, jej szerokości czy kolorów - co jest możliwe dzięki funkcji axis().

Przykład 1.7 – funkcja axis()

W przykładzie 1.7, dostosowano oś x przez dodanie koloru złotego, zmianę typu linii na przerywaną oraz dostosowanie grubości. Natomiast oś y dostosowano dodatkowo poprzez zmianę koloru wartości na osi (col.axis):

```
plot(wybraneAuta, type = "b")
axis(1, col = "gold", lty = 2, lwd = 0.5)  #lty - typ linii, lwd - grubosc linii
axis(2, lty = 2, lwd = 3, col.axis = "green") #col.axis - kolor wartosci na osi
```

Przykład 1.8 – xlim i ylim

W przykładzie 1.8, rozszerzono wykres o dostosowanie skali na osiach. Zwróć uwagę, że argumenty xlim i ylim są argumentami dla funkcji plot(), a nie dla funkcji axis():

```
plot(wybraneAuta, type = "b", xlim = c(0,15), ylim = c(0,40))
axis(1, col = "gold", lty = 2, lwd = 0.5)
axis(2, lty = 2, lwd = 3, col.axis = "green")
```

1.4 Funkcja text()

Jedną z podstawowych funkcji wykorzystywanych przy tworzeniu wykresów jest funkcja text(). Jak nazwa wskazuje pozwala na dodawanie tekstu do wykresu, np. etykiet dla punktów na wykresie. Argumentami tej funkcji poza tekstem są współrzędne punktu, w którym będzie znajdować się określony tekst. W przykładzie 1.9 do każdego punktu dodano etykietę zawierającą słowo "car" i określony numer porządkowy (np. car 1), dostosowano czcionkę używając argumentu cex, pozycję ustalono przez argument pos. Jako opisano w komentarzu w przykładzie 1.9, argument pos może przyjmować wartości z przedziału od 1 do 4, które kolejno odpowiadają ułożeniu: poniżej, po lewej, powyżej, po prawej. Jednakże nie jest to koniec możliwości tego argumentu, pos umożliwia także podanie dokładnych współrzędnych, co zostanie pokazane na dalszym przykładzie.

Przykład 1.9 – funkcja text()

```
plot(x = wybraneAuta$speed, y = wybraneAuta$dist,
     xlab = "szybkosc",
     ylab = "dystans",
     main = "Wykres szybkosc vs dystans",
     col = "blue",
     pch = 16
)
text(wybraneAuta$speed, wybraneAuta$dist,
     paste("car", c(1:10)),
     cex = 0.5,
     pos = 1, #1 - ponizej, 2 - po lewej, 3 - powyzej, 4 - po prawej
     col = "red"
)
```

Dodatkowo zamiast podawania konkretnego tekstu etykiety dla punktu, często wykorzystuje się funkcję row.names(), która pozwala na uzyskanie nazwy wierszy o ile takie są zdefiniowane.

1.5 Funkcja abline() - dorysowywanie linii

Funkcja abline() dorysowuje linię, ale nie tworzy nowego wykresu tak jak funkcja plot(). Dlatego przed wywołaniem funkcji abline() należy zapewnić okno graficzne. Dla narysowania linii poziomej - argumentem jest h, dla linii pionowej - argumentem jest v. Argumenty funkcji abline() są takie same jak w przypadku funkcji plot(), np. lty odpowiada za styl linii, lwd odpowiada za grubość linii itd. W przykładzie 1.10 do wykresu dodano linię poziomą, która wskazuje na średni dystans.

Przykład 1.10 – funkcja abline()

```
#dodaj do wykresu linie (np. oznaczajaca sredni dystans)
plot(x = wybraneAuta$speed, y = wybraneAuta$dist,
     xlab = "szybkosc",
     ylab = "dystans",
     main = "Wykres szybkosc vs dystans",
     col = "blue",
     pch = 16
)
abline(h = mean(wybraneAuta$dist), col="red") # v - wertykalnie, h - horyzontalnie
```

1.6 Funkcja points()

Poza możliwością dodawania linii do wykresu, R umożliwia dodanie punktów do wykresu poprzez wykorzystanie funkcji points(). Najczęstszym zabiegiem jest wybranie konkretnych punktów i oznaczenie ich w sposób wyróżniający od pozostałych punktów na wykresie. W przykładzie 1.11 wyselekcjonowano tylko wybrane samochody. Wybrano tylko te rekordy, w których wartości w kolumnie „speed" były większe od 9. Wykorzystano funkcję with(), która pozwala odwoływać się do kolumny ramki danych jak do zwykłej zmiennej. Po wyselekcjonowaniu punktów, można nanieść je na wykres w konkretnym kolorze i konkretnym kształcie.

Przykład 1.11 – funkcja points() + with()

```
najszybszeAuta <- with(wybraneAuta, wybraneAuta[speed > 9, ])

plot(wybraneAuta,
     xlab = "szybkosc",
     ylab = "dystans",
     main = "Wykres szybkosci",
     col = "blue",
     pch = 20
)
points(najszybszeAuta, col = "red", pch = 17)
abline(v = mean(wybraneAuta$speed), col="red") # v - wertykalnie, h - horyzontalnie
text(6, 9, "srednia szybkosc", col = "red")
```

Przykład 1.12 – funkcja points() + which()

Przykład 1.12 różni się od przykładu 1.11 tym, że dane wyselekcjonowano w trochę inny sposób, a na wykresie przedstawiono tylko parametr speed. Wykorzystano funkcję which aby wyciągnąć indeksy wyselekcjonowanych danych. Te dane tworzą współrzędne dla punktów, które zostaną naniesione na wykres.

```
najszybsze <- wybraneAuta$speed[wybraneAuta$speed > 9]
idNajszybsze<-which(wybraneAuta$speed>9)

plot(wybraneAuta$speed,
     xlab = "numer auta",
     ylab = "szybkosc",
     main = "Wykres szybkosci",
     col = "blue",
     pch = 20
)
points(idNajszybsze, najszybsze, col = "red", pch = 17)
abline(h = mean(wybraneAuta$speed), col="red") # v - wertykalnie, h - horyzontalnie
text(2.5, 8.3, "srednia szybkosc", col = "red")
```

1.7 Zmiana czcionki – cex

Zmiana czcionki jest możliwa dzięki argumentowi cex. R umożliwia zmianę czcionki różnych elementów na wykresie, m.in. tytułu (cex.main), etykiety osi (cex.lab), wartości na osi (cex.axis) itd. W poniższym przykładzie utworzono zmienną x przechowującą wartości od 0.5 do 3.0 (z krokiem 0.5), natomiast do zmiennej y przypisano wartość 1 odpowiednią ilość razy. Dla wymienionych wartości utworzono wykres i naniesiono tekst o określonym rozmiarze. Zwróć uwagę, że rozmiar tekstu jest zależny o zmiennej x. Dodatkowo w przykładzie 1.13 wykorzystano argument pos = 3, który ustala pozycję tekstu „powyżej" punktu. Natomiast w przykładzie 1.14 zamiast ustalenia wartości argumentu pos, określono dokładne współrzędne dla tekstu.

Przykład 1.13 – cex

```
x <- seq(0.5, 3, 0.5)
y <- rep(1, length(x))

plot(x, y, main = "Efekt zmiany czcionki")
text(x, y, pos = 3, labels = x, cex = x)
```

Przykład 1.14 – pos z wykorzystaniem dokładnego położenia (x,y)

```
plot(x, y, main = "Efekt zmiany czcionki", cex.main = 1.5)
text(x, y+0.1, labels = x, cex = x)
```

2 Funkcja boxplot() i attach()

Wykres pudełkowy (boxplot), czasami zwany wykresem skrzynkowym lub pudełkowym z wąsami. Pudełko jest narysowane od pierwszego do trzeciego kwartyla. Pogrubiony odcinek oznacza medianę. Natomiast wąsy są narysowane pomiędzy najmniejszą i największą obserwacją. Punkty, które nie są objęte wąsami i nie mieszczą się w pudełku są obserwacjami odstającymi. Dla zaprezentowania różnych możliwości tego typu wykresu, przygotowano 2 zestawy danych.

Przykład 2.1 - boxplot() dla pojedynczej kolumny

```
#Utworz przykladowa ramke danych - wiek przypadkowych osob
wiekPrzypadkowychLudzi <- data.frame(
    kobieta = c(25,27,30,27,35),
    mezczyzna = c(40,35,29,50,27),
    dziecko = c(6,7,10,15,8)
)
wiekPrzypadkowychLudzi

boxplot(wiekPrzypadkowychLudzi$kobieta)
abline(h = mean(wiekPrzypadkowychLudzi$kobieta), col="red")
text(1.3, 29.5, "srednia", col = "red")
abline(h = median(wiekPrzypadkowychLudzi$kobieta), col="blue")
text(1.3, 27.5, "mediana", col = "blue")
```

Na jednym wykresie można narysować pudełka dla określonej zmiennej w różnych grupach, w związku z tym przygotowano dane w trochę inny sposób niż w przykładzie powyżej. Dodatkowo zostanie tu wywołana funkcja attach(), która udostępnia nazwy kolumn (dołącza je do środowiska) i ułatwia korzystanie z danych. Umożliwia dostęp jak do zwykłych zmiennych. Funkcja detach() odłącza nazwy ze środowiska.

Przykład 2.2 - boxplot() z grupowaniem (bez użycia attach())

```
#Utworz przykladowa ramke danych
wiekPrzypadkowychLudzi2 <- data.frame(
    grupa = c("kobieta", "mezczyzna", "dziecko"),
    wiek = c(27,30,5,25,35,10,29,29,7,40,42,15)
)
wiekPrzypadkowychLudzi2

boxplot(wiekPrzypadkowychLudzi2$wiek ~ wiekPrzypadkowychLudzi2$grupa,
    xlab = "grupa",
    yab = "wiek"
)
```

Przykład 2.3 - boxplot() z grupowaniem (z użyciem attach())

```
attach(wiekPrzypadkowychLudzi2)
boxplot(wiek ~ grupa, data = wiekPrzypadkowychLudzi2)
```

Po wywołaniu attach(), dostępne są nazwy “wiek” i “grupa” bez odwoływania się przez operator $ jak w przykładzie 2.2.

Dla czytelności wykresów możemy pokolorować pudełka. R umożliwia pokolorowanie tego typu wykresu na 2 sposoby: obwódka pudełka lub wypełnienie. Zauważ, że kolory mogą być podane zgodnie z kodowaniem RGB.

Przykład 2.4 - boxplot() kolorowanie pudełek

```
boxplot(wiek ~ grupa, data = wiekPrzypadkowychLudzi2,
    border = c("#99cc00", "#660099", "#0047b3")
)

boxplot(wiek ~ grupa, data = wiekPrzypadkowychLudzi2,
    col = c("#99cc00", "#660099", "#0047b3")
)
```

2.1 Funkcja par() - mfrow i mfcol

Chcąc otworzyć wiele wykresów w jednym oknie graficznym można wykorzystać funkcję par() i argumenty mfrow() i mfcol(). W przykładzie 2.5 wykorzystano argument mfrow, który przyjmuje dwuelementowy wektor c(1,3). Pierwszy element wektora oznacza liczbę wierszy, a drugi liczbę kolumn.

Przykład 2.5 - wiele wykresów w jednym oknie

```
par(mfrow=c(1, 3)) #okreslono 3 wykresy w jednym wierszu
boxplot(wiekPrzypadkowychLudzi$kobieta)
boxplot(wiekPrzypadkowychLudzi$mezczyzna)
boxplot(wiekPrzypadkowychLudzi$dziecko)
boxplot(wiekPrzypadkowychLudzi$dziecko)
dev.off()
```

Po zakończeniu wyświetlania wielu wykresów w jednym oknie trzeba zamknąć aktywne okno graficzne, w przeciwnym razie wszystkie kolejne wykresy pojawią się obok siebie. Funkcja dev.off() zamyka bieżące, aktywne urządzenie graficzne.

3 Funkcja scatterplot()

Wykres rozrzutu Scatterplot() można traktować jak rozszerzoną wersję plot(), która jest dostępna do użycia po zainstalowaniu pakietu “car” (pamiętaj żeby po zainstalowaniu pakietu dołączyć bibliotekę). W przykładach poniżej wykorzystano dane “cars”, do których dodano jeszcze jedną kolumnę, zawierającą 2 marki samochodów. Przewaga funkcji scatterplot() nad plot() jest widoczna już w przykładzie 3.1, gdzie pojawia się także naniesiona linia regresji liniowej, dopasowana krzywa wygładzona i wykresy pudełkowe. Natomiast w przykładzie 3.2 utworzono wykres z podziałem na grupy, grupowanie po kolumnie “auto”, która została dodana do istniejących danych.

Przykład 3.1 - scatterplot()

```
wszystkieAuta <- cars
wszystkieAuta[["auto"]] <- c("BMW", "AUDI")

attach(wszystkieAuta)
scatterplot(dist ~ speed, data = wszystkieAuta, grid = FALSE)
```

Przykład 3.2 - scatterplot() z wyróżnieniem grup

```
scatterplot(dist ~ speed | auto, data = wszystkieAuta, grid = FALSE)
```

4 Funkcja dotchar()

Funkcja dotchar() umożliwia rysowanie wykresów kropkowych. W poniższych przykładach wykorzystano generowane losowo dane zawierające statystyki dotyczące kilku wybranych zawodów pomiędzy kobietami i mężczyznami. Dla utworzenia zestawienia ilości wykorzystano funkcję table().

Przykład 4.1 - dotchar() i table()

```
statystykiZawodow <- data.frame(
    plec = rep(c("kobieta", "mezczyzna"), each = 10),
    zawod = sample(c("lekarz", "prawnik", "strazak", "nauczyciel"), 20, replace = TRUE),
    wiek = sample(c(27:60), 20, replace = TRUE)
)
statystykiZawodow
attach(statystykiZawodow)

wybraneDane <- table(zawod, plec)
dotchart(wybraneDane)
```

Przykład 4.2 - scatterplot() z wyróżnieniem grup

W przykładzie 4.2 dodano kolorowanie na wykresie kropkowym. Wyróżniamy nie tylko kolor dla punktów (color), ale również kolor dla grup (gcolor) i kolor linii na wykresie (lcolor):

```
dotchart(wybraneDane,
    pch = 19,
    color="skyblue", #kolor punktow na osi
    gcolor="mediumblue", #kolor grup
    lcolor="darkblue", lwd=3 #kolor linii
)
```

5 Funkcja stripchart()

Wykres paskowy stripchart() zwany również jednowymiarowym wykresem rozrzutu jest traktowany jako alternatywa dla wykresów boxplot(), gdy reprezentowane dane są małe. Na przykładzie wcześniej utworzonych danych przygotowano wykresy paskowe w różnych orientacjach i w wariancie kolorystycznym.

Przykład 5.1 - stripchart()

```
stripchart(wiekPrzypadkowychLudzi2$wiek ~ wiekPrzypadkowychLudzi2$grupa,
    data = wiekPrzypadkowychLudzi2, pch = 16)
```

Przykład 5.2 - stripchart() z argumentem vertical

```
attach(wiekPrzypadkowychLudzi2)
stripchart(wiek ~ grupa, data = wiekPrzypadkowychLudzi2, pch = 16, vertical = TRUE)
```

Przykład 5.3 - stripchart() z kolorem

```
#dla kazdej grupy zmien ksztalt i kolor punktow
stripchart(wiek ~ grupa, data = wiekPrzypadkowychLudzi2, vertical = TRUE,
    col = c("lightblue", "mediumblue", "darkblue"),
    pch = c(15, 17, 19)
)
```

6 Funkcja pie()

Wykres kołowy można utworzyć wykorzystując funkcję pie(), jako argumenty przyjmuje dane oraz może przyjąć wektor etykiet. Inne argumenty służą manipulowaniu kolorami, etykietami, kolorami etykiet itd. Zwróć uwagę jak można manipulować etykietami. W przykładzie 6.1 jako etykiety przypisano elementy pobrane bezpośrednio z utworzonych danych, podczas gdy w przykładzie 6.2 jako etykiety przypisano wartości z nowo utworzonego wektora.

Przykład 6.1 - pie()

```
pewneStatystyki <- data.frame(
    grupa = c("lubi czekolade", "nie lubi czekolady", "brak opinii"),
    licznosc = c(86,30,9)
)
pewneStatystyki

pie(pewneStatystyki$licznosc, labels = pewneStatystyki$grupa)
```

Przykład 6.2 - pie() i heat.colors()

W przykładzie 6.2 poza innym sposobem określenia etykiet, przypisano także kolory przez funkcję heat.colors(). Funkcja ta dostosowuje kolory w zależności od liczności danych:

```
pie(pewneStatystyki$licznosc,
    main = "Jak bardzo lubimy czekolade!",
    labels = c("lubi czekolade", "nie lubi czekolady", "nie mam zdania"),
    col = heat.colors(3)
)
```

6.1 Funkcja pie() i legend()

Często tworzonym dodatkiem dla czytelności wykresów jest legenda. Niektóre typy wykresów mogą utworzyć legendę poprzez argument, inne natomiast mogą wykorzystać funkcję legend(). W poniższym przykładzie 6.3 użyto funkcji legend(). Zwróć uwagę, że pierwszy argument funkcji określa lokalizację legendy. Pozostałe argumenty określają etykiety, wielkość czcionki czy kolor.

Przykład 6.3 - funkcja legend()

```
pie(pewneStatystyki$licznosc,
    main = "Jak bardzo lubimy czekolade!",
    labels = paste("liczba osob: ", pewneStatystyki$licznosc),
    col = c("#99cc00", "#660099", "#0047b3")
)
legend("topright",
    cex = 0.9,
    c("osoby lubiace czkolade", "osoby nie lubiace czkolady", "osoby bez zdania"),
    fill = c("#99cc00", "#660099", "#0047b3"),
    title = "legenda"
)
```

7 Funkcja barplot()

Funkcja barplot() umożliwia tworzenie wykresów słupkowych w poziomie lub w pionie. Przykład 7.1 i 7.2 przedstawia ułożenie wykresów słupkowych w pionie i poziomie. Przykład 7.3 i 7.4 mają dodane kolory (dla wszystkich słupków jednocześnie oraz oddzielnie), a także pokazują inny sposób przypisania wartości dla argumentu names.arg, który opisuje słupki.

Przykład 7.1 - barplot() domyślnie

```
1 barplot(pewneStatystyki$licznosc)
```

Przykład 7.2 - barplot() horyzontalnie

```
1 barplot(pewneStatystyki$licznosc, horiz = TRUE) #wersja horyzontalna wykresu slupkowego
```

Przykład 7.3 - barplot() i names.arg

```
1 barplot(pewneStatystyki$licznosc,
2     names.arg = pewneStatystyki$grupa,
3     col = "lightpink",
4     border = "red"
5 )
```

Przykład 7.4 - barplot(), names.arg i kolory

```
1 barplot(pewneStatystyki$licznosc,
2     main = "Nastawienie do czekolady",
3     names.arg = c("lubi", "nie lubi", "brak opinii"),
4     col = c("lightpink", "lightblue", "lightgreen"),
5     border = c("red", "blue", "darkgreen")
6 )
```

7.1 Funkcja barplot(), funkcja legend() vs argument legend

Dodanie do wykresu słupkowego legendy jest możliwe na 2 sposoby. Pierwszy z nich z użyciem funkcji legend(), drugi z użyciem argumentu legend. Przy użyciu argumentu nie trzeba dostosowywać kolorów, tak jak w przypadku funkcji legend().

Przykład 7.5 - barplot() i funkcja legend()

```
1  barplot(pewneStatystyki$licznosc,
2      main = "Nastawienie do czekolady",
3      col = c("lightpink", "lightblue", "lightgreen"),
4      border = c("red", "blue", "darkgreen")
5  )
6  legend("topright",
7      legend = c("lubi", "nie lubi", "brak opinii"),
8      fill = c("lightpink", "lightblue", "lightgreen"),
9      title = "Legenda:"
10 )
```

Przykład 7.6 - barplot() i argument legend

```
barplot(pewneStatystyki$licznosc,
    main = "Nastawienie do czekolady",
    col = c("lightpink", "lightblue", "lightgreen"),
    border = c("red", "blue", "darkgreen"),
    legend = c("lubi", "nie lubi", "brak opinii")
)
```

8 Funkcja hist()

Innym rodzajem wykresu jest histogram hist(), poniżej przedstawiono użycie funkcji na przykładzie losowego zbioru 100 ocen.

Przykład 8.1 - hist()

```
oceny <- sample(c(2,3,3.5,4,4.5,5), 100, replace = TRUE)
oceny
head(oceny, 10)

hist(oceny, col = "skyblue")
```

9 Zapisywanie wykresów

Do zapisu wykresów do pliku wykorzystuje się różne funkcje (w zależności o preferowanego formatu), m.in.: pdf(), png(), jpeg(), svg(). Poniżej przykład generowania wykresów w formacie pdf i png.

Przykład 9.1 - zapisywanie wykresu jako .pdf

```
pdf("Wykres szybkosc vs dystans.pdf")
    plot(x = wybraneAuta$speed, y = wybraneAuta$dist,
    xlab = "szybkosc",
    ylab = "dystans",
    main = "Wykres szybkosc vs dystans",
    col = "blue", #zmiana koloru punktow, funkcja colors() pokazuje dostepne kolory
    pch = 16 #ksztalt punktow, zobacz ?pch
)
dev.off()
```

Przykład 9.2 - zapisywanie wykresu jako .jpg

```
jpeg("Wykres szybkosc vs dystans.jpg", width = 600, height = 400)
    plot(x = wybraneAuta$speed, y = wybraneAuta$dist,
    xlab = "szybkosc",
    ylab = "dystans",
    main = "Wykres szybkosc vs dystans",
    col = "blue", #zmiana koloru punktow, funkcja colors() pokazuje dostepne kolory
    pch = 16 #ksztalt punktow, zobacz ?pch
)
dev.off()
```

10 Zadania

Rozwiąż poniższe zadania, a utworzony skryptu .R zostaw do okazania pod koniec zajęć.

Zad. 1

Sprawdź jakie wartości przyjmuje argument pch i wykorzystaj wykres z przykładu 1.2, aby zmienić kształt punktów na trójkąty o kolorze zielonym.

Zad. 2

Wykorzystaj wykres z przykładu 1.4 i przetestuj inne typy: p, b, h, s. Jakim typom wykresów odpowiadają wymienione wartości argumentu type?

Zad. 3

Wykorzystaj wykres z przykładu 1.6 i przetestuj inne typy pudełka: "l", "7", "c", "u", "]", co zaobserwowałeś, jaki jest kształt?

Zad. 4

Wykorzystaj wykres z przykładu 1.10 i dodaj zieloną, wyraźną linię przerywaną, wyznaczoną w punkcie średniej szybkości.

Zad. 5

Wykorzystaj wykres z przykładu 1.12, aby dodatkowo oznaczyć na nim samochody wyróżniające się najmniejszą szybkością. Oznacz je fioletowym kwadratem.

Zad. 6

Sprawdź jakie argumenty przyjmuje funkcja boxplot() następnie wykorzystaj dane z przykładu 2.3 i utwórz wykres pudełkowy dla tych danych w postaci horyzontalnej.

Zad. 7

Wykorzystaj funkcję par() i argument mfrow, aby w 1 oknie graficznym utworzyć 2 wykresy: pierwszy wykres boxplot(), a drugi stripchart(). Wykresy mają zawierać informacje o zakresie wieku dla każdej badanej grupy (wykorzystaj dane: wiekPrzypadkowychLudzi2).

Zad. 8

Znajdź informacje jaki pakiet należy doinstalować i jakiej funkcji użyć, aby utworzyć wykres kołowy 3D. Utwórz wykres kołowy 3D dla danych "pewneStatystyki" z przykładu 6.1.

Zad. 9

Wykorzystaj wykres z przykładu 6.3 i dodaj etykiety. Etykiety powinny zawierać informacje w procentach, tzn. ile procent osób lubi/ nie lubi/ nie ma zdania na temat czekolady.

Zad. 10

Przygotuj dowolne dane, możesz użyć funkcji sample() czy rep(), następnie wygeneruj dowolnie wybrany przez siebie wykres i zapisz go w formacie jpg.