

install.packages('PogromcyDanych')

library(PogromcyDanych)

setLang(lang = 'eng')

library(ggplot2)

# zad.1 (a) parametr method ustawiony na lm wskazuje model liniowej regresji, mozna sprawdzic

# samo geom\_smooth()

cats\_birds %>%

ggplot(aes(x=length,

y=speed,

color=group)) +

geom\_point() +

geom\_smooth(method = lm)

# zad.1 (b)

library(PogromcyDanych)

setLang(lang = 'eng')

cats\_birds

cats\_birds %>%

ggplot(aes(x=length,

y=speed,

color=group)) +

geom\_point() +

geom\_smooth(method = lm)

# zad.2 se= FALSE sprawia, że nie ma pasa (przedzialu ufnosci) wokol linii regresji

pearson %>%

ggplot(aes(x = son,

y = father)) +

geom\_point() +

geom\_smooth(method = lm, se = FALSE) #

# zad.3

data.series <- seriesIMDB

# Wykres jest mało czytelny ze wzgledu na duza ilosc seriali (ponad 200)

data.series %>%

ggplot(aes(x=note,

y=series)) +

geom\_boxplot()+

theme(axis.text = element\_text(size=2))

# Gdy wezmiemy mniejsza probke jest czytelniejszy. Dla niektorych serialow

# moglo byc malo probek lub tylko jedna, wtedy "pudelko" sprowadza sie do kreski

# oznaczajacej mediane

data.series %>%

sample\_n(size = 100, replace = F) %>%

ggplot(aes(x=note,

y=series)) +

geom\_boxplot()+

theme(axis.text = element\_text(size=4))

# zmieniamy kolejnosc seriali porzodkujac ze wzgledu na rosnoca mediane

data.series$series <- reorder(data.series$series, data.series$note, median)

# ...i dla zmodyfikowanych danych tworzymy boxplota

data.series %>%

ggplot(aes(x=note,

y=series)) +

geom\_boxplot()+

theme(axis.text = element\_text(size=2))

# zad. 4

diagnosis -> diag

View(diag)

diag %>%

ggplot(aes(x=eduk4\_2013,

fill= gp29))+

geom\_bar(position='stack',

color='black')

# zad. 5

auta2012 %>%

filter(Brand == 'Volkswagen', Model == 'Passat') %>%

ggplot(aes(x= Year,

y = Price))+

geom\_point() +

geom\_smooth()

auta2012 %>%

filter(Brand == 'Volkswagen', Model == 'Passat') %>%

ggplot(aes(x= Year,

y = Price))+

geom\_point() +

geom\_smooth(method = lm)

?geom\_smooth()

auta2012 %>%

filter(Brand == 'Volkswagen', Model == 'Passat') %>%

ggplot(aes(x= Year,

y = Price))+

geom\_point() +

geom\_smooth(method = 'loess')

# zad. 6

# kod startowy

ggplot(cats\_birds, aes(x = weight, y = speed, size = lifespan, color = lifespan)) +

geom\_point()

# http://sape.inf.usi.ch/quick-reference/ggplot2/shape#:~:text=Geoms%20that%20draw%20points%20have%20a%20%22shape%22%20parameter.,to%20127%20correspond%20to%20the%20corresponding%20ASCII%20characters.

# wartosci parametru shape

ggplot(cats\_birds, aes(x = weight,

y = speed,

size = lifespan,

color = lifespan)) +

geom\_point(shape=18)+

scale\_color\_gradient(low = 'green', high = 'red')+

xlab("weight") +

ylab("speed") +

ggtitle("Zależność wagi od prędkości")

# zad. 7

#W pierwszym kroku przygotowujemy dane, tworzymy kolumne czestotliwosci

# w kolejnym tworzymy barplota, a nastepnie poleceniem coord\_polar tworzymy pie chart

auta2012

auta2012 %>%

filter(Brand == 'Toyota') %>%

group\_by(Model) %>%

summarise(`most popular`=n()) %>%

arrange(desc(`most popular`)) %>%

head(5) -> toyotas

toyotas <- mutate(toyotas, `freq`= `most popular`/ sum(`most popular`))

toyotas %>%

ggplot(aes(x = "", y=freq, fill = factor(Model))) +

geom\_bar(width = 1, stat = "identity") +

theme(axis.line = element\_blank(),

plot.title = element\_text(hjust=0.5)) +

labs(fill="Model",

x=NULL,

y=NULL,

title="Pie Chart of models",

caption="Source: auta2012") +

coord\_polar(theta = "y", start=0)

# zad. 8

ggplot(pearson, aes(x = son)) +

geom\_histogram(aes(y = ..density..),

colour = 1, fill = "white") +

geom\_density() ->p1

ggplot(pearson, aes(x = father)) +

geom\_histogram(aes(y = ..density..),

colour = 1, fill = "white") +

geom\_density() -> p2

plot\_grid(p1, p2)

# zad. 9

iris %>%

ggplot(aes(x=Sepal.Length,

y=Sepal.Width,

color=Species,

shape=Species)) +

geom\_point()+

geom\_density2d()+

ggtitle("Iris dataset")

# zad.10

#Pierwsza metoda - generujemy każdy z wykresów z osobna, a następnie tworzymy subplota

iris %>%

ggplot( aes(x=Petal.Length, y=Petal.Width))+

geom\_point(colour='green')+

geom\_smooth()+

ggtitle('setosa') -> p1

iris %>%

filter(Species== 'setosa') %>%

ggplot( aes(x=Petal.Length, y=Petal.Width))+

geom\_point(colour='red')+

geom\_smooth()+

ggtitle('setosa') -> p2

iris %>%

filter(Species== 'virginica') %>%

ggplot( aes(x=Petal.Length, y=Petal.Width))+

geom\_point(colour='blue')+

geom\_smooth()+

ggtitle('virginica') -> p3

iris %>%

filter(Species== 'versicolor') %>%

ggplot( aes(x=Petal.Length, y=Petal.Width))+

geom\_point(colour='yellow')+

geom\_smooth()+

ggtitle('versicolor') -> p4

plot\_grid(p1, p2, p3, p4)

#Druga metoda - modyfikujemy zbiór dodając osobną klasę ALL

#(kopiujemy cały zbiór i dodajemy nową klasę ALL i dołączamy ten zbiór starego)

iris.pom <- iris

iris.pom$Species = 'ALL'

iris2 = rbind(iris.pom, iris)

iris2

#Generujemy wykres za pomocą warstwy facet

iris2 %>%

ggplot(aes(x=Petal.Length,

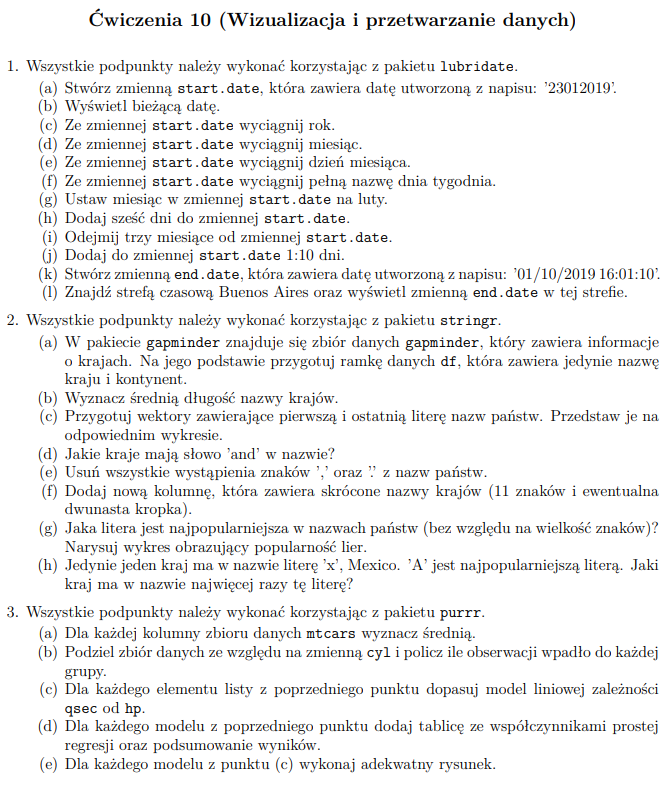
y= Petal.Width,

color = Species))+

geom\_point() +

geom\_smooth()+

facet\_wrap(. ~Species, scales='free')



library(tidyverse)

library(lubridate)

install.packages('tidyverse')

#zad.1

#a)

start.date <- fast\_strptime(x = '23012019', format = "%d%m%Y")

start.date

#b)

today()

#c)

year(start.date)

#d)

month(start.date)

#e)

day(start.date)

#f)

wday(ymd(start.date))

wday(ymd(start.date), label=TRUE)

#g)

month(start.date) <- 2

start.date

#h)

d <- day(start.date) + 6

day(start.date) <- d

start.date

#i)

m <- month(start.date)

m <- m - 3

month(start.date) <- m

start.date

#j)

ymd(start.date) + days(c(1,2,3,4,5,6,7,8,9,10))

#k)

end.date <- fast\_strptime(x = '01/10/2019 16:01:10', format = "%d/%m/%Y %H:%M:%S")

end.date

#l)

Sys.timezone()

# Different time zones

OlsonNames()

# How many different time zones

OlsonNames() %>% length()

with\_tz(end.date, 'America/Argentina/Buenos\_Aires')

# Zad. 2

#a)

library(tidyverse)

library(stringr)

install.packages('gapminder')

library(gapminder)

gapminder %>%

select(country, continent) -> df

#b)

df %>%

mutate(`average length`= str\_length(country)) %>%

select(`average length`) %>%

map\_dbl(.x = ., .f = mean)

#c)

x <- 'Remaining'

nchar(x)

substr(x, nchar(x), nchar(x))

df %>%

select(country) %>%

mutate(`last\_letter` = substr(country, str\_length(country),str\_length(country)),

`first\_letter` = substr(country, 1,1)) -> df1

l <- df1$last\_letter[]

f <- df1$first\_letter[]

l

f

#d)

countries <- df1$country[]

indx <- str\_detect(string = countries, pattern = "and") # returns TRUE / FALSE

countries[indx]

countries[grepl(pattern = "and", x = countries)]

#e)

countries1 <- str\_replace\_all(string = countries, pattern = "[.,]", replacement = "")

View(countries1)

#f)

# str\_sub() - Extract part of s string

# similar base R: substr()

countries.short <- str\_sub(string = countries, start=1, end=11)

df1 <- df

df1['short'] <- countries.short

df1 <- mutate(df1, `len` = str\_length(df1$short))

View(df1)

df1$short <- paste(df1$short, ".", sep='')

# g)

countries.df

countries.df %>%

mutate(letter.a = str\_count(country, 'n'))

for (letter in letters)

{

countries.df[letter] <- str\_count(countries.df$country, letter)

}

countries.df

#we can use countries.df to create a plot....

#Zad.3

library(tidyverse)

library(purrr)

library(dplyr)

install.packages('dplyr')

#a)

mtcars %>%

map\_dbl(.x = ., .f = mean)

#b)

mtcars %>%

group\_by(cyl) %>%

count()

n.mtcars <- mtcars %>%

group\_by(cyl) %>%

nest()

#c)

df.mtcars.models <- n.mtcars %>%

mutate(model = map(.x = data, # fit a linear model with map

.f = ~lm(qsec~hp,

data = .)))

df.mtcars.models

#d)

model\_4 <- df.mtcars.models %>% filter(cyl==4) %>% pull(model)

model\_4

model\_4 %>% flatten()

model\_4 %>% flatten() %>% names()

model\_4 %>% flatten() %>% pluck(coefficients)

model\_4 %>% flatten() %>% pluck(coefficients) %>% enframe()

model\_4 %>% flatten() %>% pluck(coefficients) %>% .[[1]] # intercept a0

model\_4 %>% flatten() %>% pluck(coefficients) %>% .[[2]] # a1

model\_4 %>% map(summary)

## extract model summary

model\_4 %>% map(summary) %>% map\_dbl("r.squared")

#Podobnie robim dla cyl==6 i cyl ==8

model\_6 <- df.mtcars.models %>% filter(cyl==6) %>% pull(model)

model\_8 <- df.mtcars.models %>% filter(cyl==8) %>% pull(model)

df.mtcars.models

df.mtcars.models[[1]]

df.mtcars.models[[2]]

df.mtcars.models[[3]]

df.mtcars.models$data[[2]]

#e)

df.mtcars.models %>%

select(cyl, data) %>%

unnest(cols = c(data)) %>%

ungroup() %>%

# create a scarret plot

ggplot(aes(x = qsec,

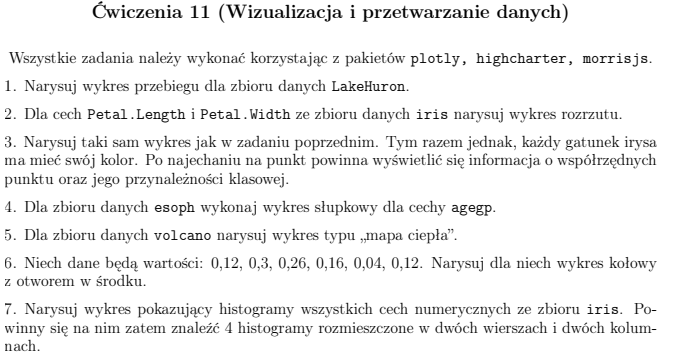
y = hp,

color = as.factor(cyl))) +

geom\_point() +

geom\_smooth(method = lm, se=F)

facet\_wrap(. ~ cyl)



#cw. 11

## zad. 1

# 1

LakeHuron

library(plotly)

years = seq(1875, 1972,1)

v\_result <- plot\_ly(x = years,

y = LakeHuron)

v\_result

years = seq(1875, 1972,1)

v\_result <- plot\_ly(x = years,

y = LakeHuron,

type = 'scatter',

mode = 'lines')

v\_result

var\_a <- list(

autotick = FALSE,

ticks = "outside", #"outside"

tick0 = 0,

dtick = 5,

ticklen = 6,

tickwidth = 3,

tickcolor = toRGB("red")

)

v\_result <- v\_result %>% layout(xaxis = var\_a, yaxis = var\_a)

v\_result

v\_result <- plot\_ly(x = years,

y = LakeHuron,

type = 'scatter',

mode = 'lines',

line = list(color = 'Red',

width = 3,

dash = 'dash')) #'dot'

v\_result

## Zad.2

iris

v\_result <- plot\_ly(x = ~iris$Petal.Length, y = ~iris$Petal.Width, mode='markers')

v\_result

## Zad. 3

v\_result <- plot\_ly(data = iris, x = ~Petal.Length, y = ~Petal.Width, color = ~Species)

v\_result

v\_result <- plot\_ly(data = iris, x = ~Petal.Length, y = ~Petal.Width, mode='markers',

text = ~paste("Petal.Length: ",

Petal.Length,

'<br>Petal.Width:', Petal.Width,

'<br>Species:', Species),

color = ~Species)

v\_result

## Zad.4

library(tidyverse)

esoph %>%

group\_by(agegp) %>%

summarise(agegp, `amount`= n()) -> esoph.data

View(esoph.data)

v\_result <- plot\_ly(data = esoph.data, x = esoph.data$agegp, y = esoph.data$amount, type = 'bar')

v\_result

#Zad. 5

?volcano

fig <- plot\_ly(z = volcano, type = "heatmap")

fig

m <- matrix(rnorm(9), nrow = 3, ncol = 3)

fig <- plot\_ly(

x = c("a", "b", "c"), y = c("d", "e", "f"),

z = m, type = "heatmap"

)

fig

fig <- plot\_ly(z = volcano, colors = "Greys", type = "heatmap")

fig

# zad. 6

#How to create pie chart

var1 <- c(0.12, 0.3, 0.26, 0.16, 0.04, 0.12)

var2 <- c("India", "United States", "Canada", "China", "Poland", "Germany") #Labels

v\_data <- data.frame(var1, var2)

v\_result <- plot\_ly(v\_data, labels = ~var2, values = ~var1, type = 'pie' )

v\_result

# customization

var1 <- c(15, 50, 10, 25)

var2 <- c("India", "United States", "Canada", "China")

v\_color <- c("orange", "yellow","pink", "brown")

v\_data <- data.frame(var1, var2)

v\_result <- plot\_ly(v\_data,

labels = ~var2,

values = ~var1,

type = 'pie',

textposition = 'inside',

textinfo = 'label+percent',

insidetextfont = list(color = 'black'),

showlegend = FALSE,

marker = list(colors = v\_color)#,hole=0.4

)

v\_result

fig\_2 <- plot\_ly(v\_data,

labels = ~var2,

values = ~var1,

type = 'pie',

textposition = 'inside',

textinfo = 'label+percent',

insidetextfont = list(color = 'black'),

showlegend = FALSE,

marker = list(colors = v\_color),hole=0.4

)

fig\_2