Lista 2

Eksploracja danych

Igor Misterowicz, 282245

2025-04-29

Contents

1 Zadanie 1

1.1 a.

Pracujemy na danych iris

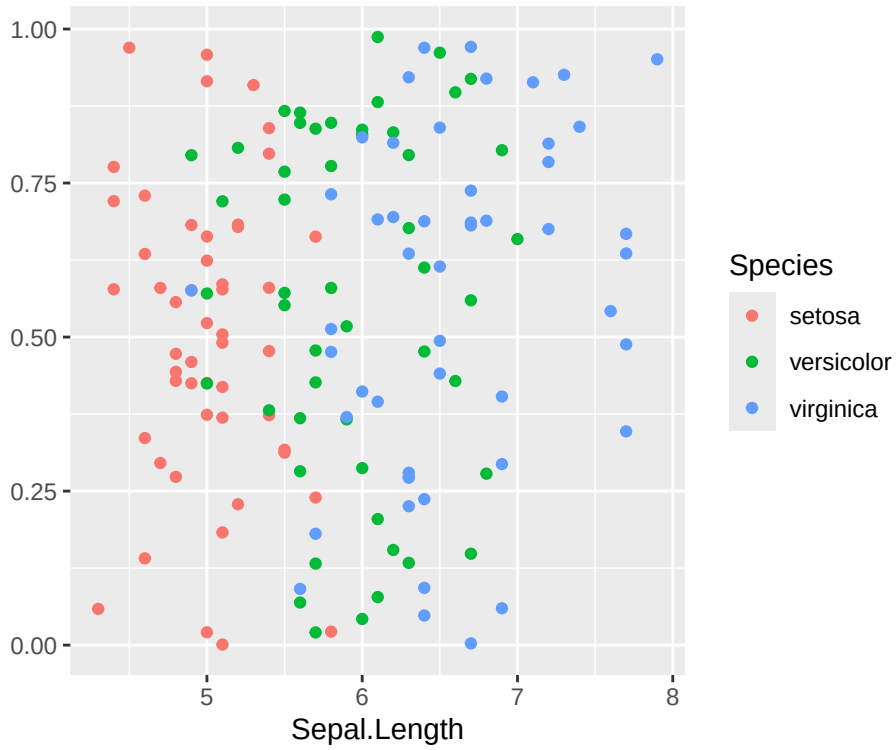
```
attach(iris)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

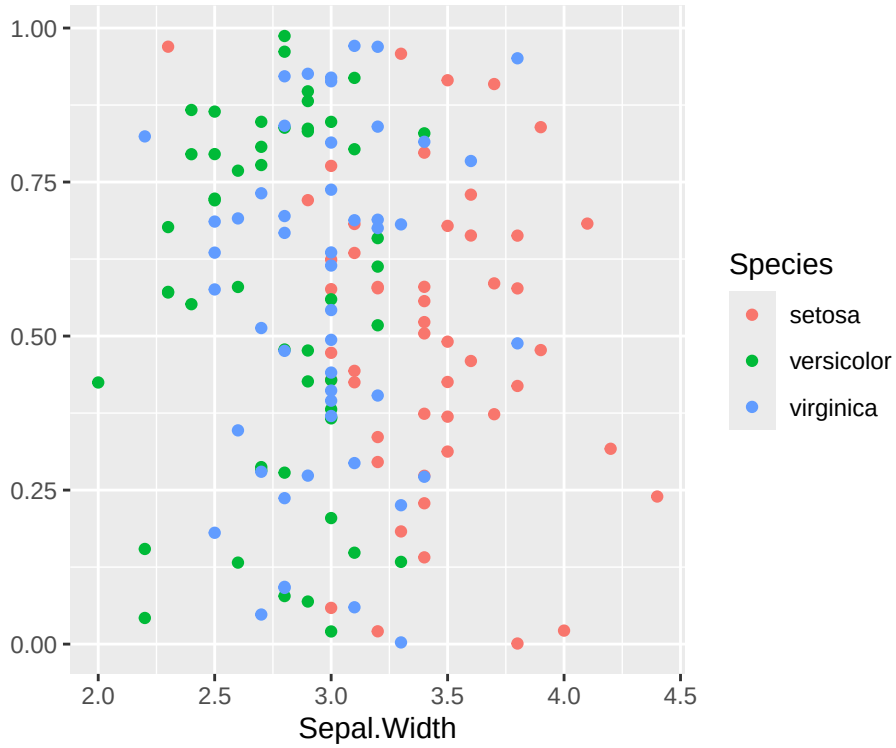
1.2 b.

Szukam cech o najlepszych i najgorszych zdolnościach dyskryminacyjnych

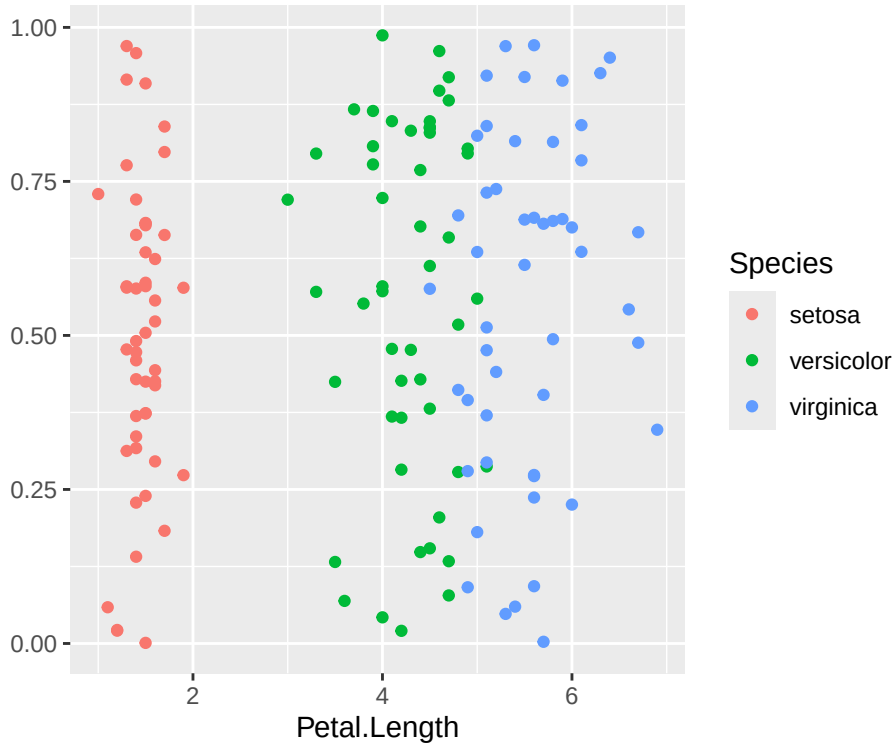
```
y_ax = runif(nrow(iris))

ggplot(iris, aes(x = Sepal.Length, y = y_ax, color = Species)) +
  geom_point() + ylab("")
```



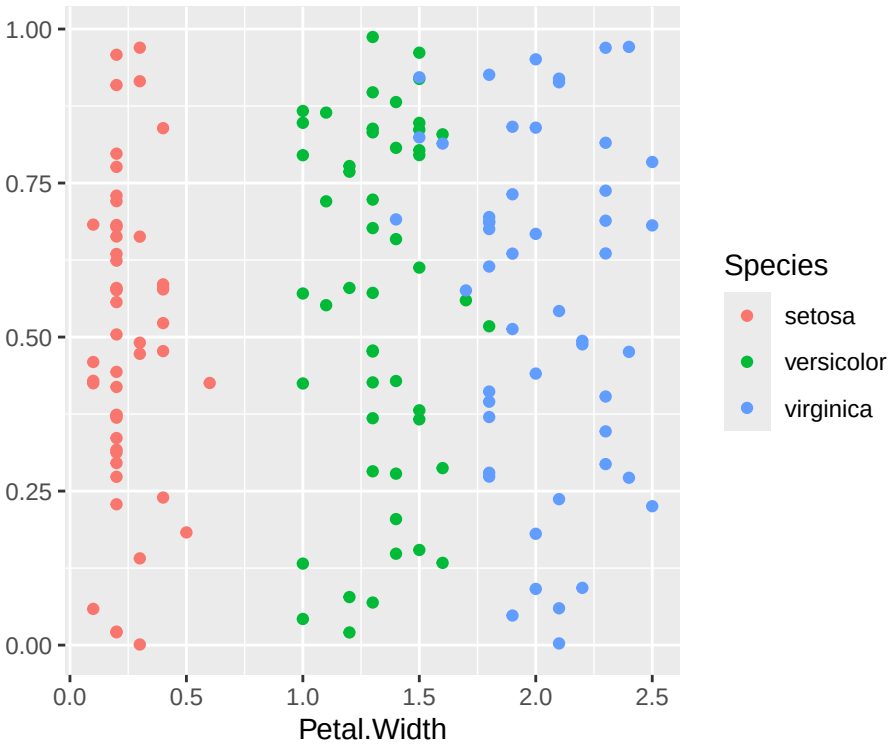

```
ggplot(iris, aes(x = Sepal.Width, y = y_ax, color = Species)) +
  geom_point() + ylab("")
```



```
ggplot(iris, aes(x = Petal.Length, y = y_ax, color = Species)) +
  geom_point() + ylab("")
```




```
ggplot(iris, aes(x = Petal.Width, y = y_ax, color = Species)) +
  geom_point() + ylab("")
```



Wizualnie dane dobrze różnicują zmienne Sepal, natomiast gorzej Petal. Patrzymy teraz na wariancje.

```
apply(iris, 2,FUN = var)
```

```
## Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
##    0.6856935    0.1899794    3.1162779    0.5810063           NA
```

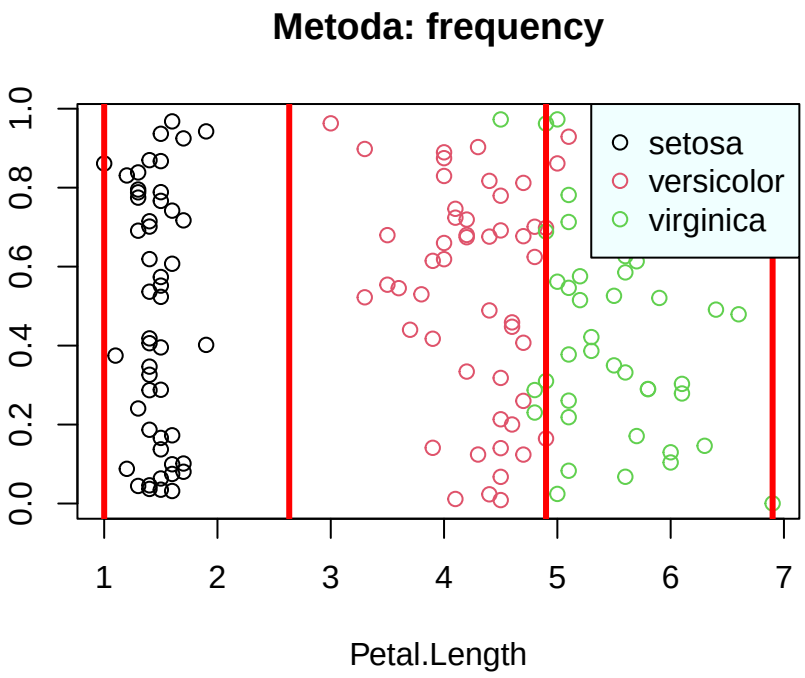
Największą wariancję ma Petal.Length, a najmniejszą Sepal.Width. Tą pierwszą możemy uznać za zmienną o najlepszych, a drugą o najgorszych zdolnościach dyskryminacyjnych.

1.3 c.

Testujemy różne algorytmy dyskretyzacji nienadzorowanej

```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Petal.Length"]
x.disc.equal.freq <- discretize(x, breaks = 3,method = "frequency")
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: frequency", xlab = "Petal.Length",
```

```
     ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3,
       pch=21, bg = "azure")
```

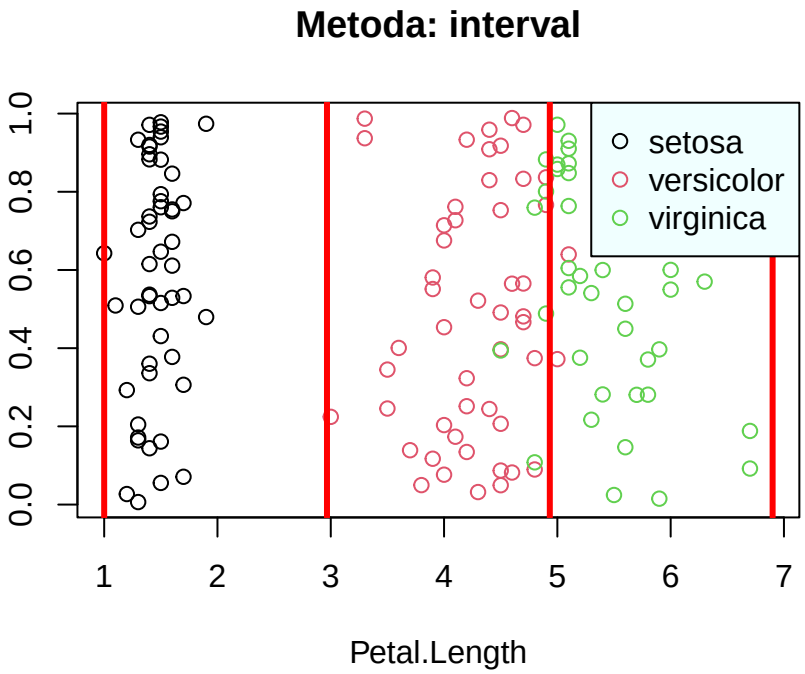



```
matchClasses(table(discretize(Petal.Length, method = "frequency", breaks = 3)
                   , iris$Species))
```

```
## Cases in matched pairs: 95.33 %

##     [1,2.63)   [2.63,4.9)    [4.9,6.9]
##     "setosa" "versicolor"  "virginica"
```

```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Petal.Length"]
x.disc.equal.freq <- discretize(x, breaks = 3, method = "interval")
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: interval",xlab = "Petal.Length",
     ylab = "")
```

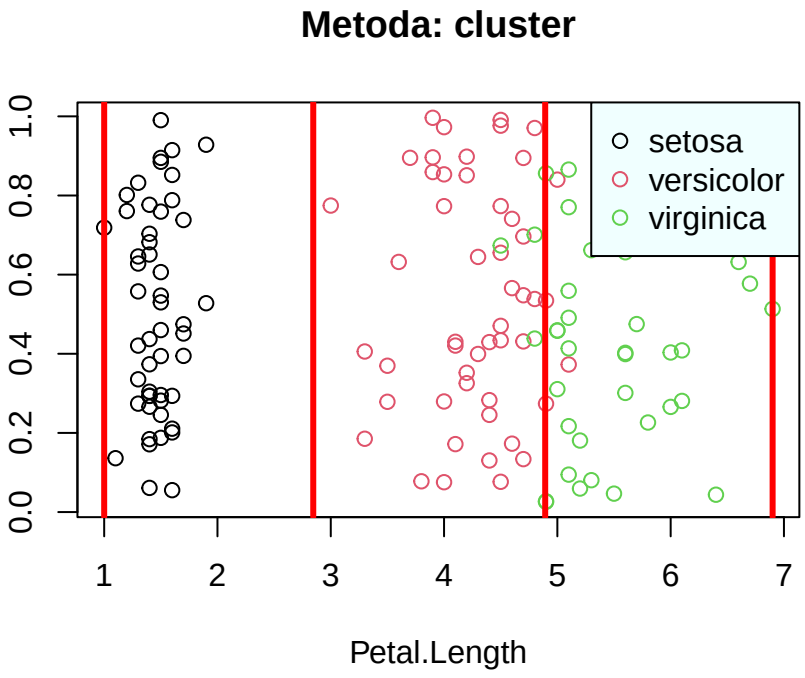
```
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```




```
matchClasses(table(discretize(Petal.Length, method = "interval", breaks = 3)
                   , iris$Species))
```

```
## Cases in matched pairs: 94.67 %

##     [1,2.97)  [2.97,4.93)   [4.93,6.9]
##     "setosa" "versicolor"  "virginica"
```

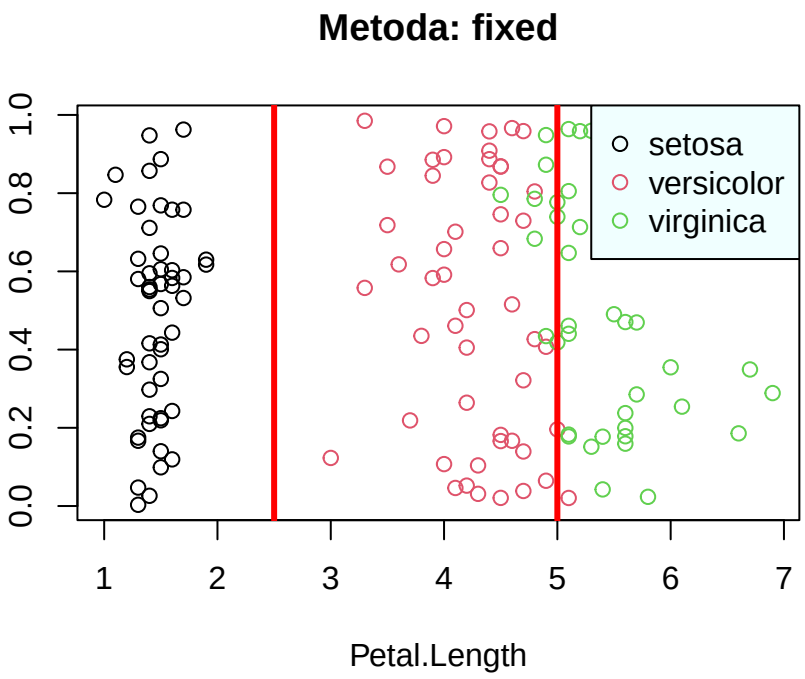
```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Petal.Length"]
x.disc.equal.freq <- discretize(x, breaks = 3, method = "cluster")
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: cluster",xlab = "Petal.Length",
     ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```




```
matchClasses(table(discretize(Petal.Length, method = "cluster", breaks = 3)
                   , iris$Species))
```

```
## Cases in matched pairs: 89.33 %

##     [1,2.95)  [2.95,5.13)   [5.13,6.9]
##     "setosa" "versicolor"  "virginica"
```

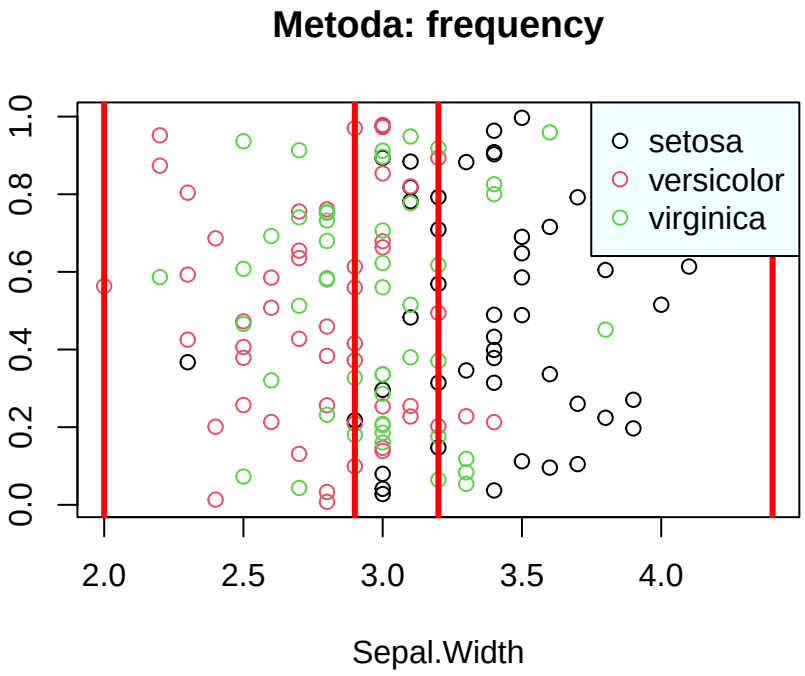
```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Petal.Length"]
x.disc.equal.freq <- discretize(x, method = "fixed",
                                breaks = c(-Inf, 2.5, 5, Inf))
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: fixed",xlab = "Petal.Length",
     ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```




```
matchClasses(table(discretize(Petal.Length, method = "fixed",
                              breaks = c(-Inf, 2.5, 5, Inf))
                   , iris$Species))
```

```
## Cases in matched pairs: 94.67 %

##   [-Inf,2.5)      [2.5,5)     [5, Inf]
##     "setosa" "versicolor"  "virginica"
```

```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Sepal.Width"]
x.disc.equal.freq <- discretize(x, breaks = 3,method = "frequency")
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: frequency",xlab = "Sepal.Width",
     ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```

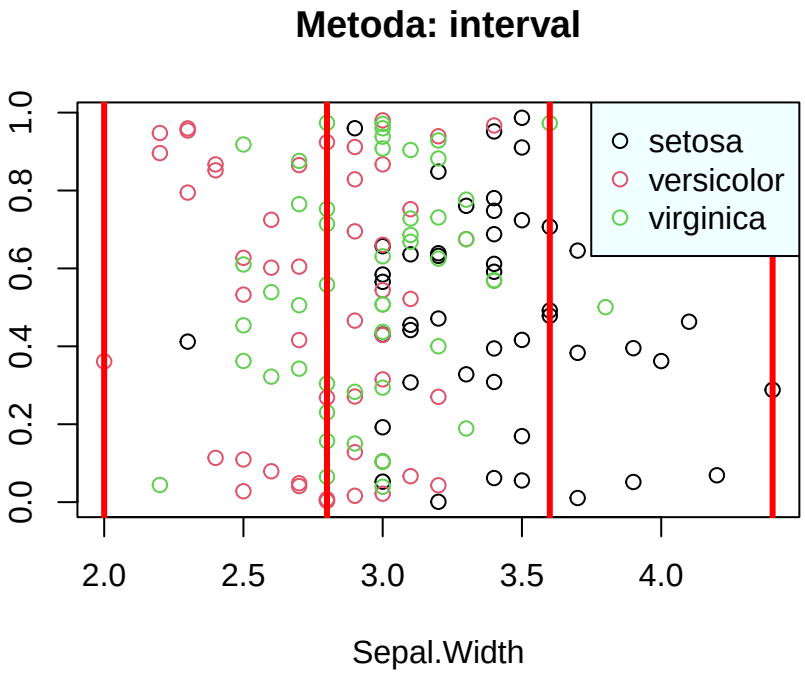



```
matchClasses(table(discretize(Sepal.Width, method = "frequency", breaks = 3)
                   , iris$Species))
```

```
## Cases in matched pairs: 55.33 %
```

```
##      [2,2.9)    [2.9,3.2)    [3.2,4.4]
## "versicolor" "versicolor"     "setosa"
```

```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Sepal.Width"]
x.disc.equal.freq <- discretize(x, breaks = 3, method = "interval")
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: interval",
     xlab = "Sepal.Width", ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```

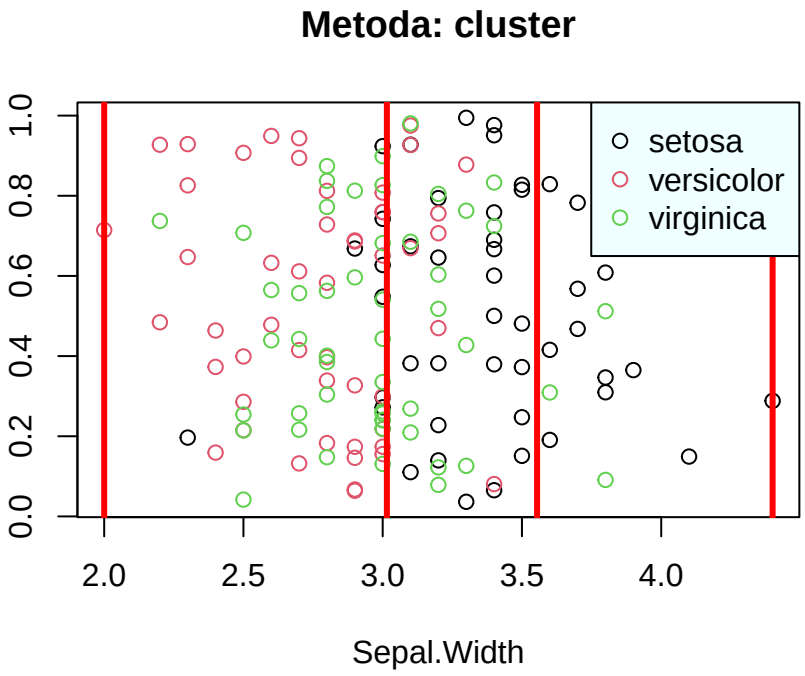



```
matchClasses(table(discretize(Sepal.Width, method = "interval", breaks = 3)
                   , iris$Species))
```

```
## Cases in matched pairs: 50.67 %
```

```
##      [2,2.8)    [2.8,3.6)    [3.6,4.4]
## "versicolor"     "setosa"     "setosa"
```

```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Sepal.Width"]
x.disc.equal.freq <- discretize(x, breaks = 3, method = "cluster")
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: cluster",xlab = "Sepal.Width",
     ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```

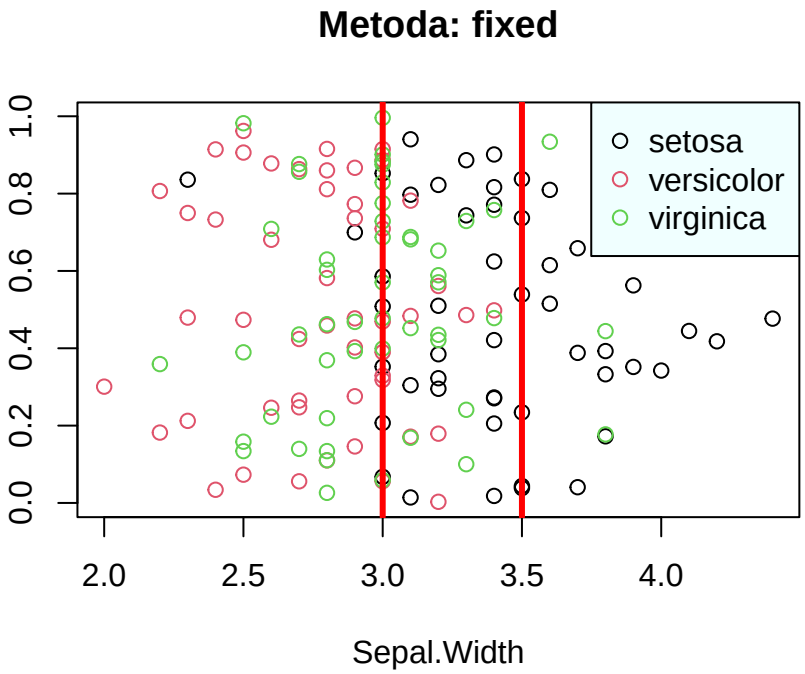



```
matchClasses(table(discretize(Sepal.Width, method = "cluster", breaks = 3)
                   , iris$Species))
```

```
## Cases in matched pairs: 56.67 %
```

```
##     [2,2.75)  [2.75,3.29)   [3.29,4.4]
## "versicolor"  "virginica"     "setosa"
```

```
n <- nrow(iris)
y <- runif(n)

x <- iris[,"Sepal.Width"]
x.disc.equal.freq <- discretize(x, method = "fixed",
                                breaks = c(-Inf, 3, 3.5, Inf))
breaks.equal.frequency <- attributes(x.disc.equal.freq)$"discretized:breaks"
plot(x, y, col=iris$Species, main = "Metoda: fixed",xlab = "Sepal.Width",
     ylab = "")
abline(v = breaks.equal.frequency, col = "red", lwd=3)
legend(x = "topright", legend=levels(iris$Species), col=1:3, pch=21,
       bg = "azure")
```




```
matchClasses(table(discretize(Sepal.Width, method = "fixed",
                              breaks = c(-Inf, 3, 3.5, Inf))
                   , iris$Species))
```

```
## Cases in matched pairs: 54.67 %
```

```
##     [-Inf,3)      [3,3.5)   [3.5, Inf]
## "versicolor"     "setosa"     "setosa"
```

Wybrany algorytm nie ma większego znaczenia, skuteczności są podobne. Odstaje jedynie metoda "interval" w przypadku zmiennej Sepal.width.

2 Zadanie 2

2.1 a.

Analiza danych jakości życia

2.2 b.

Wczytuję dane

```
p <- "C:/Users/igorm/Programowanie/data_mining/list2_files/uaScoresDataFrame.csv"
data <- read.csv(p)
```

Zera wpisane w zmienne numeryczne możemy uznać za brakujące dane. Zastępuje je średnią wartością pięciu najbliższych sąsiadów.

```
data[data == 0] <- NA
data<-kNN(data,variable=colnames(data),k=5,imp_var = FALSE)
```

Tworzę ramkę składającą się z samych zmiennych ilościowych.

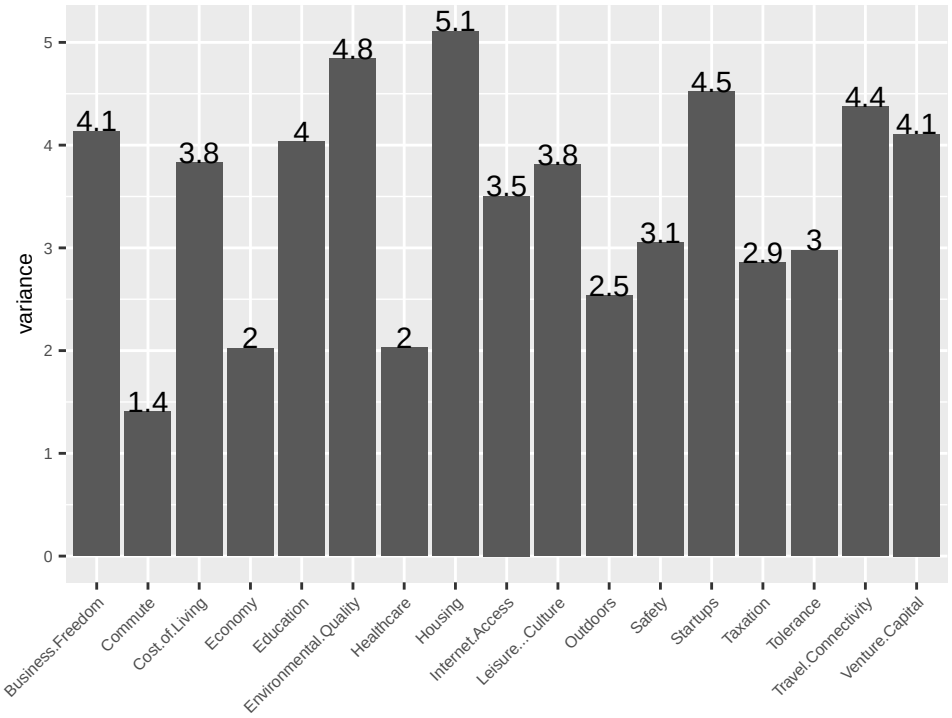
```
types <- function(d) {
data.frame(names(d), sapply(d, class))
}
types(data)
```

```
##                                    names.d. sapply.d..class.
## X                                         X          integer
## UA_Name                             UA_Name        character
## UA_Country                       UA_Country        character
## UA_Continent                   UA_Continent        character
## Housing                             Housing          numeric
## Cost.of.Living               Cost.of.Living          numeric
## Startups                           Startups          numeric
## Venture.Capital             Venture.Capital          numeric
## Travel.Connectivity     Travel.Connectivity          numeric
## Commute                             Commute          numeric
## Business.Freedom           Business.Freedom          numeric
## Safety                               Safety          numeric
## Healthcare                       Healthcare          numeric
## Education                         Education          numeric
## Environmental.Quality Environmental.Quality          numeric
## Economy                           Economy          numeric
## Taxation                         Taxation          numeric
## Internet.Access               Internet.Access          numeric
## Leisure...Culture         Leisure...Culture          numeric
## Tolerance                       Tolerance          numeric
## Outdoors                         Outdoors          numeric
```

```
num_data <- subset(data, select = c(-UA_Country,-UA_Continent,-UA_Name,-X))
```

Obliczam wariancje

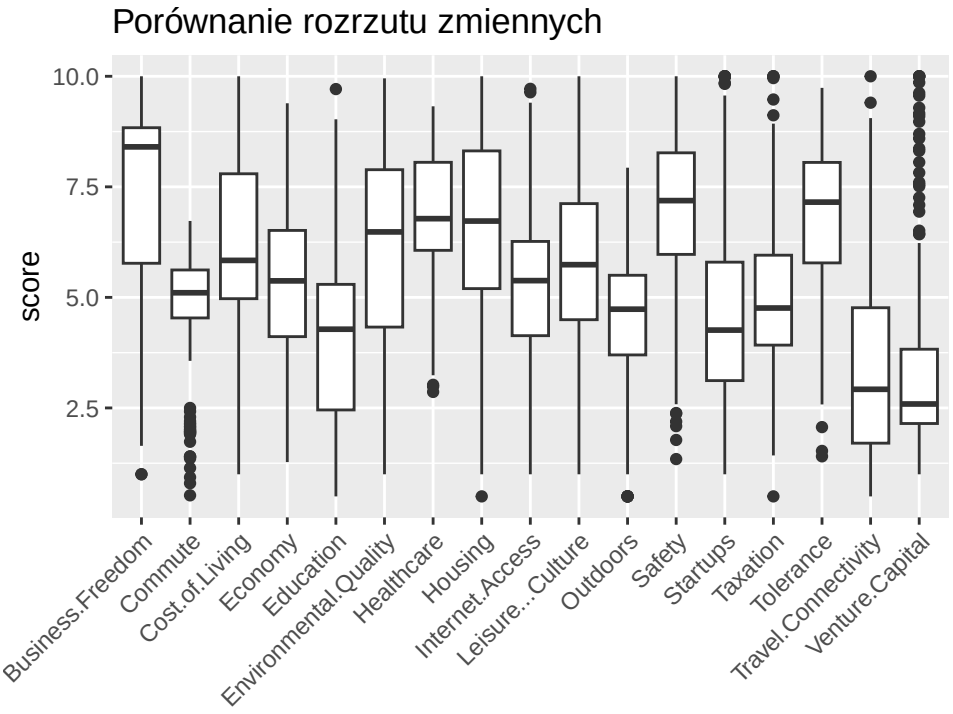
```
df <- apply(num_data,2, FUN = var)
df <- as.data.frame(df)

variance <- df[,1]
variable <- rownames(df)

ggplot(NULL, aes(x = variable, y= variance)) + geom_col() +
  geom_text(label = round(variance,1), vjust = 0) +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, hjust = 1),
        text = element_text(size = 8)) +
  xlab("")
```



Największą wariancję ma Housing, a najmniejszą Commute.

Porównuję teraz rozrzuty na wykresie pudełkowym

```r
df_long <- pivot_longer(
  num_data,
  cols = colnames(num_data),
  names_to = "column",
  values_to = "score"
)


ggplot(df_long, aes(x = column, y = score)) +
  geom_boxplot() +
  xlab("Variable") + ylab("Score") +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  xlab("") +
  ylab("score") +
  ggtitle("Porównanie rozrzutu zmiennych",)
```

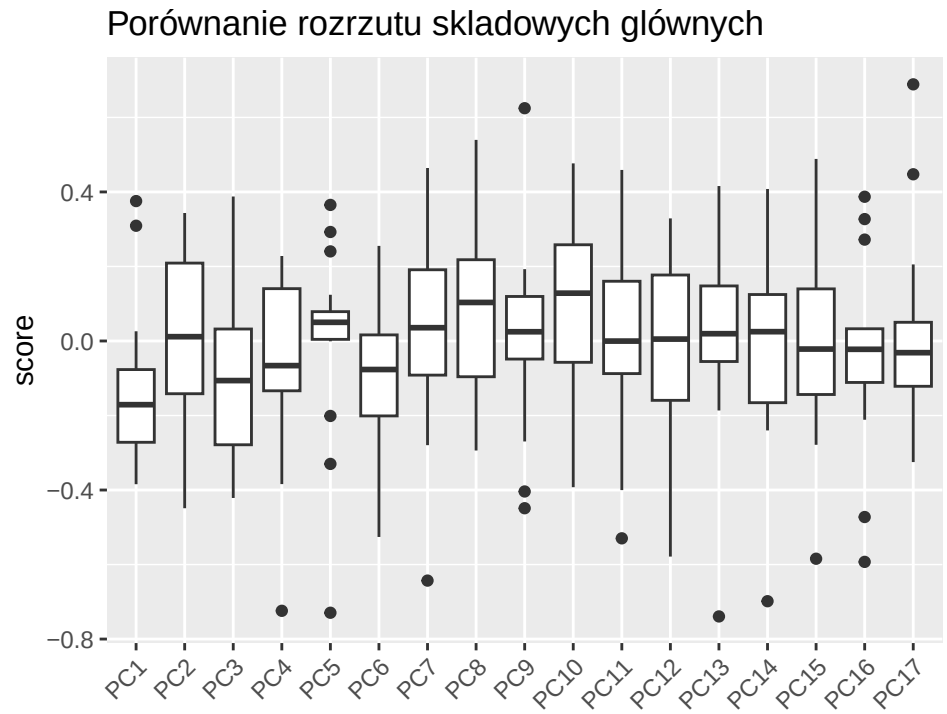


Różnice w wariancjach i średnich są znaczące, co sugeruje konieczność standaryzacji danych. Najbardziej oddalone od siebie wartości średnie mają Business.freedom oraz Venture.capital.

2.3 c.

Składowe główne oraz ich wykresy pudełkowe

```r
scaled <- as.data.frame(scale(num_data))

pca_result <- prcomp(scaled)

pca_long <- pivot_longer(
  as.data.frame(pca_result$rotation),
  cols = colnames(as.data.frame(pca_result$x)),
  names_to = "column",
  values_to = "score"
)

ord <- rep("PC",17)
for(i in 1:17){
  ord[i] <- paste0(ord[i],i)
}

pca_long$column <- factor(pca_long$column, levels = ord)


ggplot(pca_long, aes(x = column, y = score)) +
  geom_boxplot() +
  xlab("Variable") + ylab("Score") +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  xlab("") +
  ylab("score") +
  ggtitle("Porównanie rozrzutu składowych głównych",)
```



Trzy pierwsze składowe

```
pca_df <- as.data.frame(subset(pca_result$rotation, select = c(PC1,PC2,PC3)))
pca_df
```

```
##                                PC1         PC2         PC3
## Housing                 0.30929176  0.15516291 -0.09946248
## Cost.of.Living          0.37544538 -0.02246666 -0.10627945
## Startups               -0.16278004 -0.44884300 -0.15618489
## Venture.Capital        -0.17096620 -0.41635120 -0.18821238
## Travel.Connectivity    -0.19887088 -0.04246575 -0.42128190
## Commute                -0.09287197  0.28275329 -0.40602225
## Business.Freedom       -0.36938832  0.07819445  0.16515169
## Safety                 -0.04147765  0.34341435 -0.33366759
## Healthcare             -0.27950894  0.30232821 -0.15959502
## Education              -0.38437829 -0.04186491 -0.05763983
## Environmental.Quality  -0.32424720  0.20905740  0.17260250
## Economy                -0.25590908 -0.17141445  0.38772223
## Taxation                0.02603953  0.09312977  0.03210819
## Internet.Access        -0.27189547  0.01151303  0.12098979
## Leisure...Culture      -0.07671857 -0.29370787 -0.36687393
## Tolerance              -0.19184952  0.33684500 -0.03614680
## Outdoors               -0.08668809 -0.14153906 -0.27850855
```

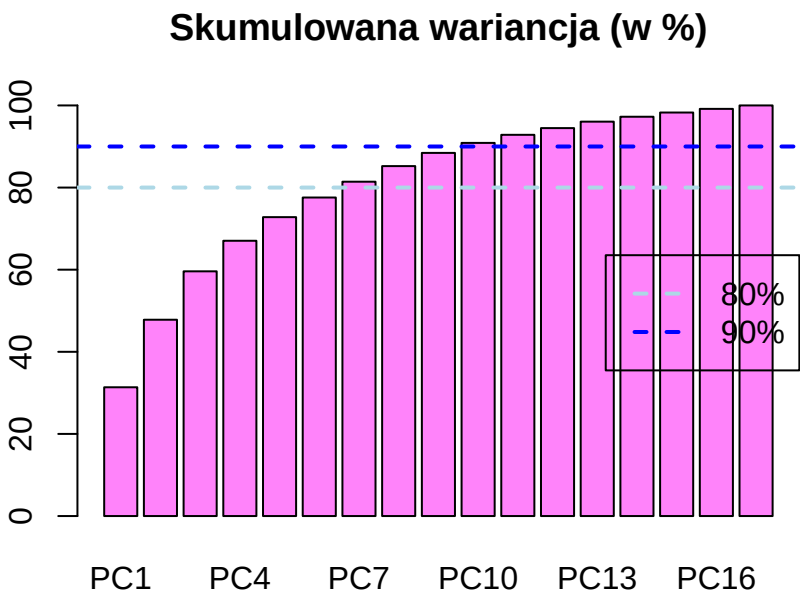
Zmienne o największej wadze

- w PC1
  - dodatnie:
    * Cost.of.Living
    * Housing
  - ujemne:
    * Business.freedom
    * Education
  - Reprezentuje niskie koszta życia oraz słaby potencjał do zarobków
- W PC2
  - dodatnie:
    * Safety
    * Tolerance
  - ujemne:
    * Startups
    * Venture.capital
  - Reprezentuje bezpieczeństwo oraz niesprzyjające warunki dla małych przedsiębiorstw
- W PC3
  - dodatnie:
    * Economy
  - ujemne:
    * Commute
    * Travel.connectivity
  - Reprezentuje zdrowie lokalnej gospodarki oraz niski poziom udogodnień

2.4 d.

Procentowy udział składowych w wariancji

```
variance <- 100*(pca_result$sdev^2)/sum(pca_result$sdev^2)
cumulative.variance <- cumsum(variance)


barplot(cumulative.variance, main="Skumulowana wariancja (w %)",
names.arg=paste0("PC",1:17),col="orchid1")
abline(h=80, col="lightblue", lty=2, lwd=2)
abline(h=90, col="blue", lty=2, lwd=2)
legend("right", legend=c("80%","90%"),
       lwd=2, lty=2, col=c("lightblue","blue"),)
```

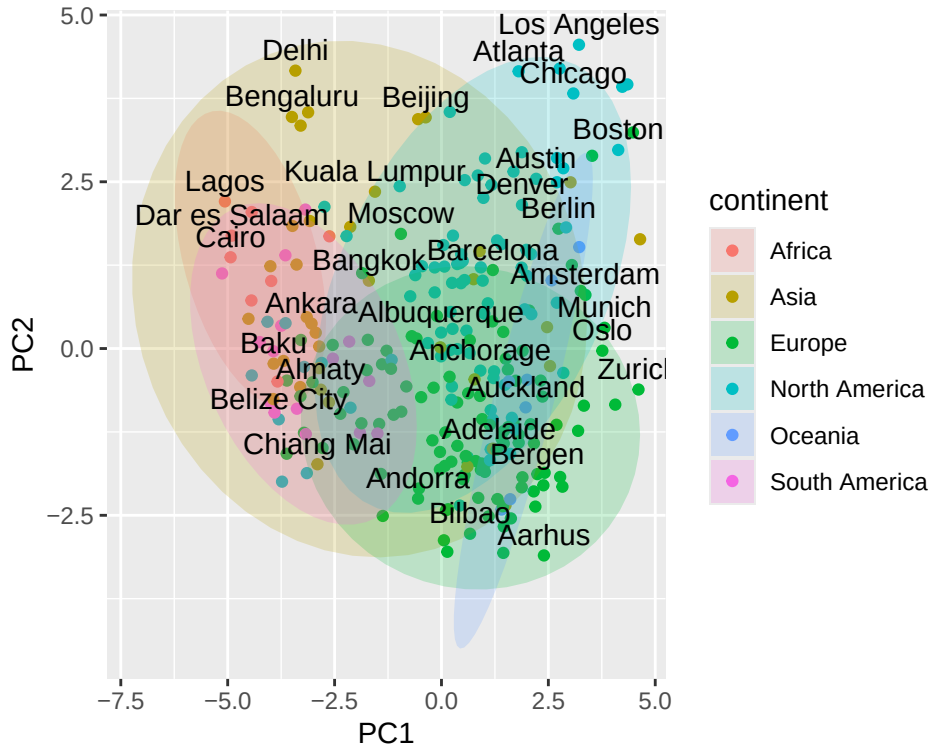


7 pierwszych zmiennych odpowiada za 80%, a 11 pierwszych za 90% wariancji.

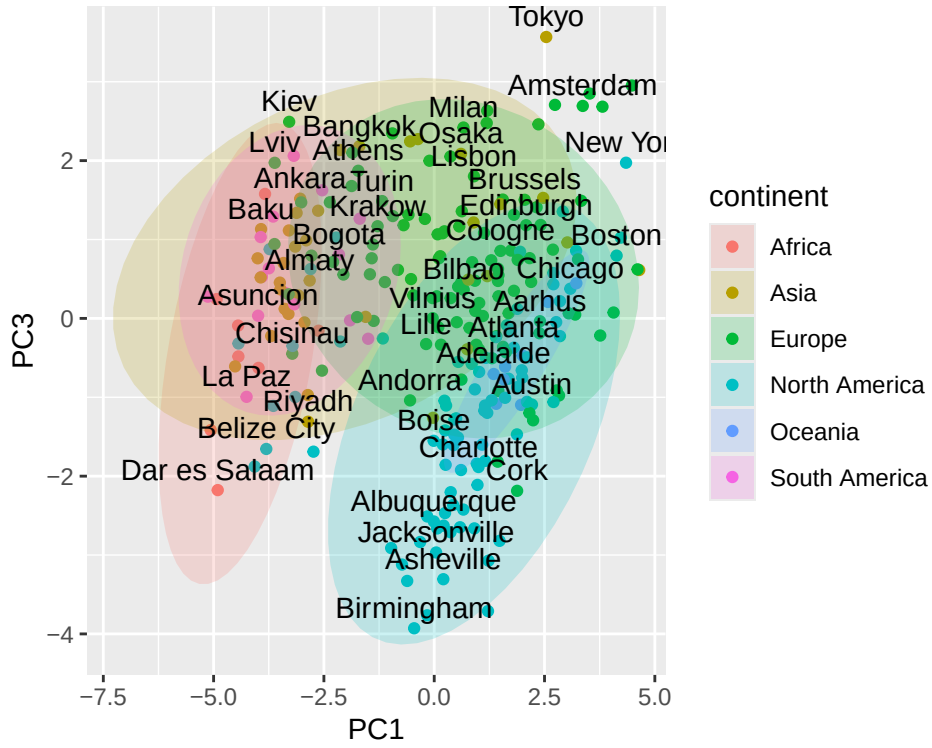
2.5 e.

Wizualizacja składowych głównych

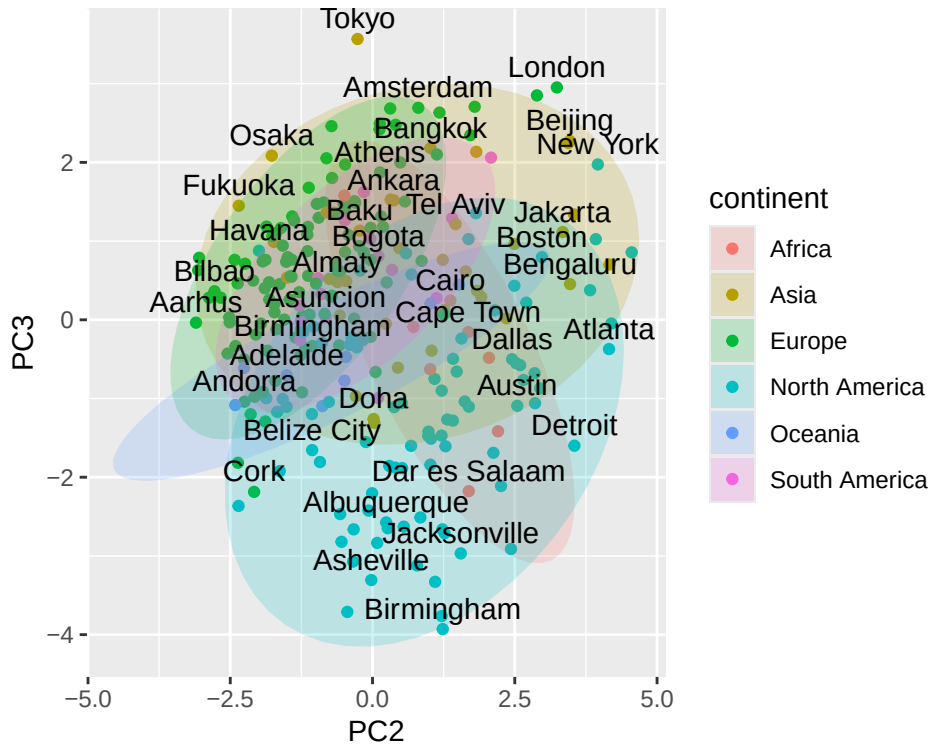
```
temp_df <- data.frame(PC1 = -pca_result$x[, 1],
                      PC2 = -pca_result$x[, 2],
                      PC3 = -pca_result$x[, 3],
                      city = data$UA_Name,
                      country = data$UA_Country,
                      continent = data$UA_Continent)



ggplot(temp_df,aes(x = PC1, y = PC2, colour = continent)) + geom_point() +
  stat_ellipse(aes(fill = continent), geom = "polygon", alpha = 0.2,
               colour = NA) +
    geom_text(aes(label = city, vjust = -0.5),check_overlap = TRUE,
              color = "black")
```




```
ggplot(temp_df,aes(x = PC1, y = PC3, colour = continent)) + geom_point() +
  stat_ellipse(aes(fill = continent), geom = "polygon", alpha = 0.2,
               colour = NA) +
  geom_text(aes(label = city, vjust = -0.5),check_overlap = TRUE,
            color = "black")
```




```
ggplot(temp_df,aes(x = PC2, y = PC3, colour = continent)) + geom_point() +
  stat_ellipse(aes(fill = continent), geom = "polygon", alpha = 0.2,
               colour = NA) +
  geom_text(aes(label = city, vjust = -0.5),check_overlap = TRUE,
            color = "black")
```



Szukam miast o ekstremalnych wartościach składowych głównych

```
cities <- rbind(temp_df[order(temp_df$PC1), ][1,],
                temp_df[order(-temp_df$PC1), ][1,],
                temp_df[order(temp_df$PC2), ][1,],
                temp_df[order(-temp_df$PC2), ][1,],
                temp_df[order(temp_df$PC3), ][1,],
                temp_df[order(-temp_df$PC3), ][1,],
                deparse.level = 0)

cities
```

```
##             PC1        PC2         PC3        city     country     continent
## 53   -5.1313842  1.1256673  0.27555053     Caracas   Venezuela South America
## 227   4.6457932  1.6382709  0.61312132   Singapore   Singapore          Asia
## 1     2.3968488 -3.1017707 -0.03630468      Aarhus     Denmark        Europe
## 140   3.2164484  4.5540295  0.85676778 Los Angeles  California North America
## 31   -0.4549556  1.2338794 -3.92862959  Birmingham     Alabama North America
## 245   2.5391124 -0.2628686  3.56743978       Tokyo       Japan          Asia
```

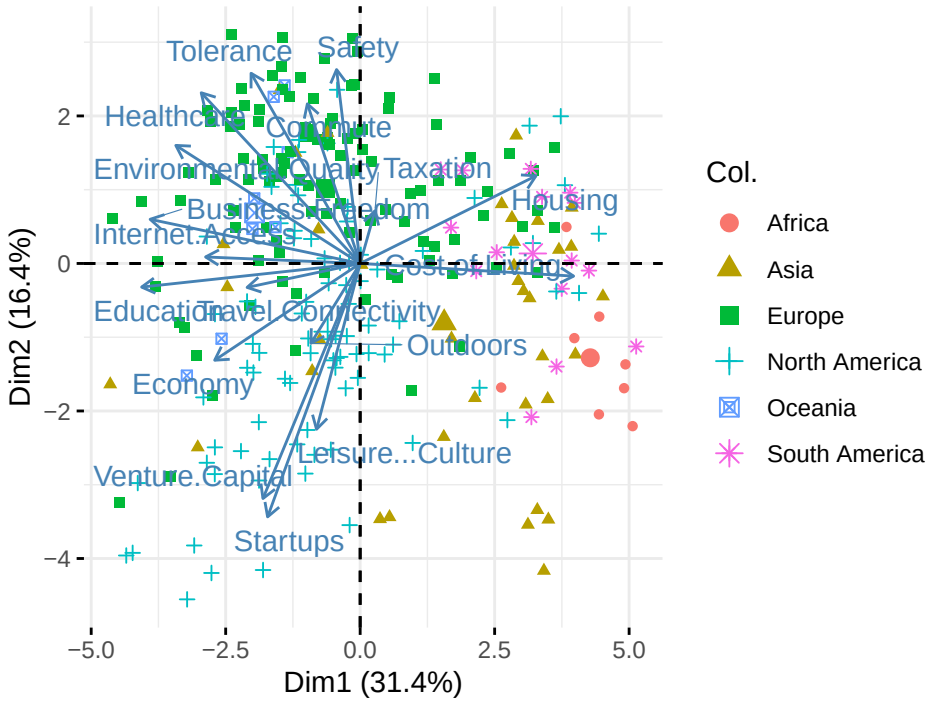
Możemy odczytać kraj oraz kontynent miast odstających od reszty.

- Największy oraz najmniejszy wskaźnik PC1 mają Singapur oraz Caracas.
- Największy oraz najmniejszy wskaźnik PC2 mają Los Angeles oraz Aarhus.
- Największy oraz najmniejszy wskaźnik PC3 mają Tokyo oraz Birmingham.

2.6 f.

Wykres korelacji na biplocie

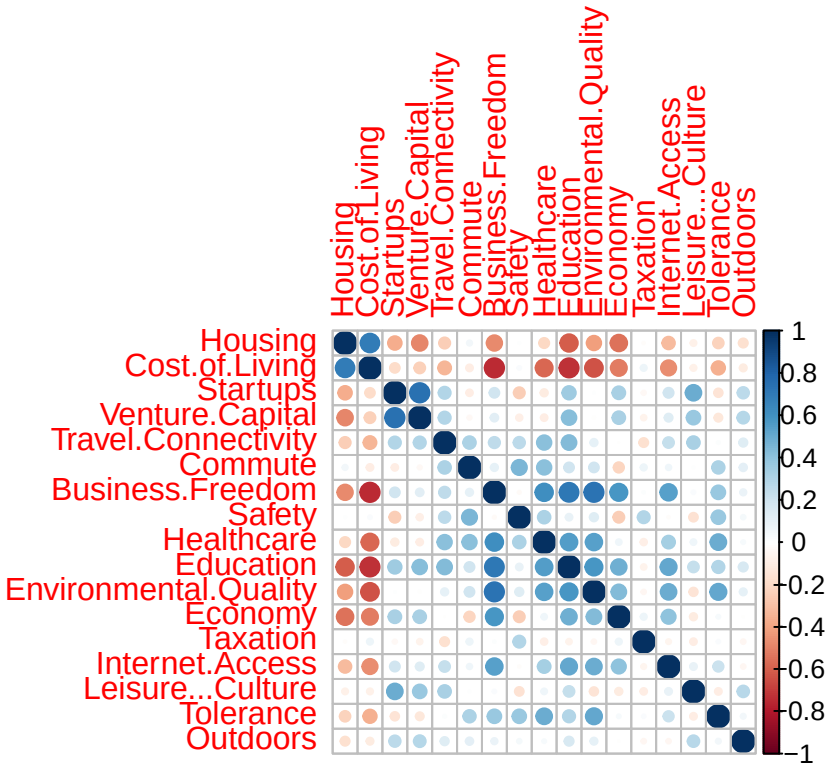
```
fviz_pca_biplot(pca_result, label  = "var", col.ind = data$UA_Continent,
                repel = T, title = "")
```



Housing i Cost.of.living mają wyraźnie inny kierunek od reszty zmiennych, pododnie w przypadku Venture.Capital i Startups.

Macierz korelacji

```
correlation.matrix <- cor(num_data)
corrplot(correlation.matrix)
```



Zauważamy następujące grupy korelacj:

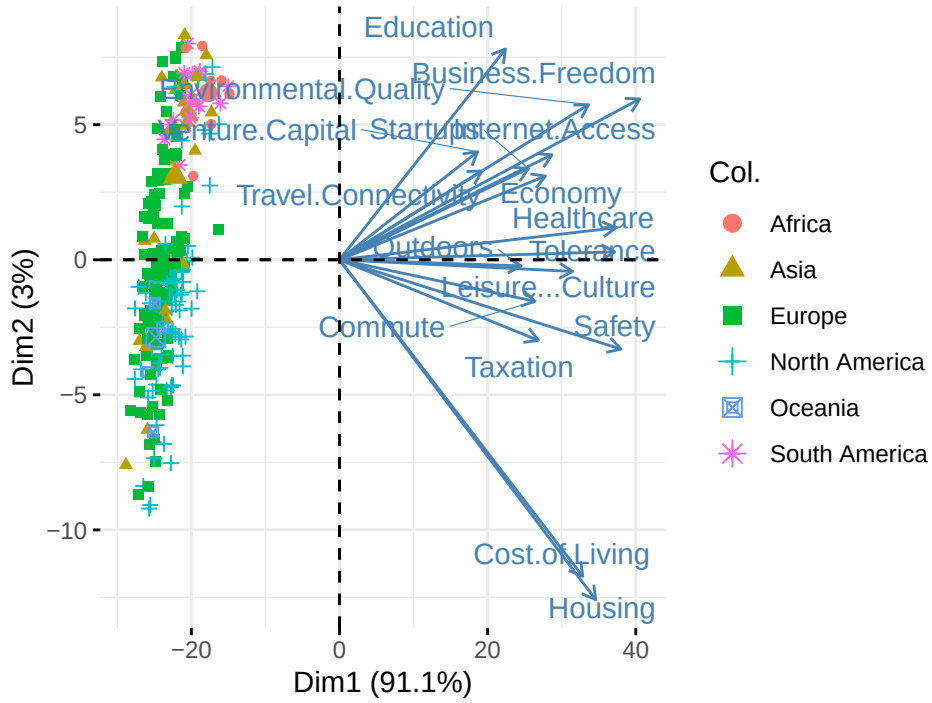
- Cost.of.living, Housing
- Business.freedom, Helathcare, Education
- Venture.capital, Startups

Wszystie trzy grupy są do siebie negatywnie skorelowane.

2.7 g.

Wykres korelacji na biplocie bez standaryzacji

```
pca_result <- prcomp(num_data,center = F)
fviz_pca_biplot(pca_result, label  = "var", col.ind = data$UA_Continent,
                repel = T, title = "")
```



Ewidentnie standaryzacja była konieczna.

Dają się zauważyć pewne zależności między naszymi składowymi głównymi, a kontynentami. Otzymujemy wyniki zgodne z intuicją tzn:

- niski potencjał do zarobków w Afryce, Azji oraz Ameryce Południowej (PC1)
- tolerancja i bezpieczeństwo w Europie, jednak mniej korzystne warunki dla startupów (PC2)
- duża ilość udogodnień w Europie (PC3)

Są jednak pewne miasta, które nie wpasowują się w powyższe zależności. Przykładem jest Singapur z rekordowym wskaźnikiem PC1. Wbrew pozorom w Ameryce Północnej wysokie finansowanie startupów ma tylko kilka odstających miast np Los Angeles.

Pierwsze trzy komponenty składowe reprezentują trzy najważniesze grupy korelacji oraz odpowiadają za 60% całej zmienności. Wobec tego naszą reprezentację możemy uznać za zadawalającą.

3 zadanie 3

3.1 a.

Analiza danych z pakietu titanic

3.2 b.

Wczytuję dane i zmieniam typy

```
data <- titanic_train
str(data)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  "" "C85" "" "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

```
data <- na.omit(data)
data$Sex <- as.factor(data$Sex)
data$Embarked <- as.factor(data$Embarked)
data$Survived <- sapply(data$Survived,
                        FUN = function(x) ifelse(x == 1, "Yes", "No"))
data$Survived <- as.factor(data$Survived)
data$Pclass <- as.ordered(data$Pclass)
Pclass <- data$Pclass
data <- subset(data, select = c(-PassengerId,-Pclass,-Name, -Ticket,-Cabin))
```

Zapisuję jednak zmienną Pclass, ponieważ będzie ona potrzebna w dalszej części analizy

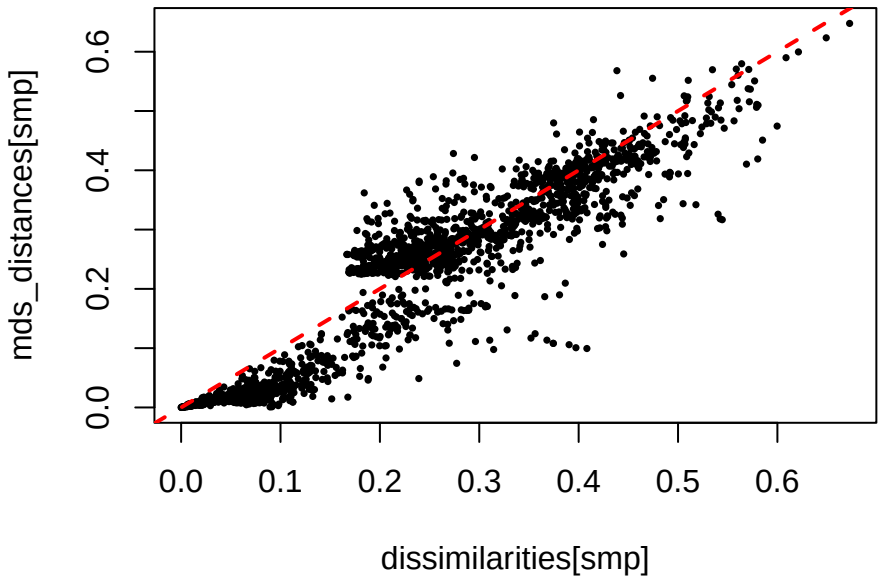
3.3 c.

Tworzę macierz odmienności

```
dissimilarities <- daisy(subset(data, select = c(-Survived)), stand=T)
dis.matrix <- as.matrix(dissimilarities)
```

Wizualizacja jakości odwzorowania

```r
mds.results <- cmdscale(dis.matrix, k=2)
mds_distances <- dist(mds.results)

smp <- sample(1:length(dissimilarities),2000)
plot(dissimilarities[smp],mds_distances[smp], pch = 16, cex = 0.5)
abline(coef=c(0,1), col="red", lty=2, lwd=2)
```

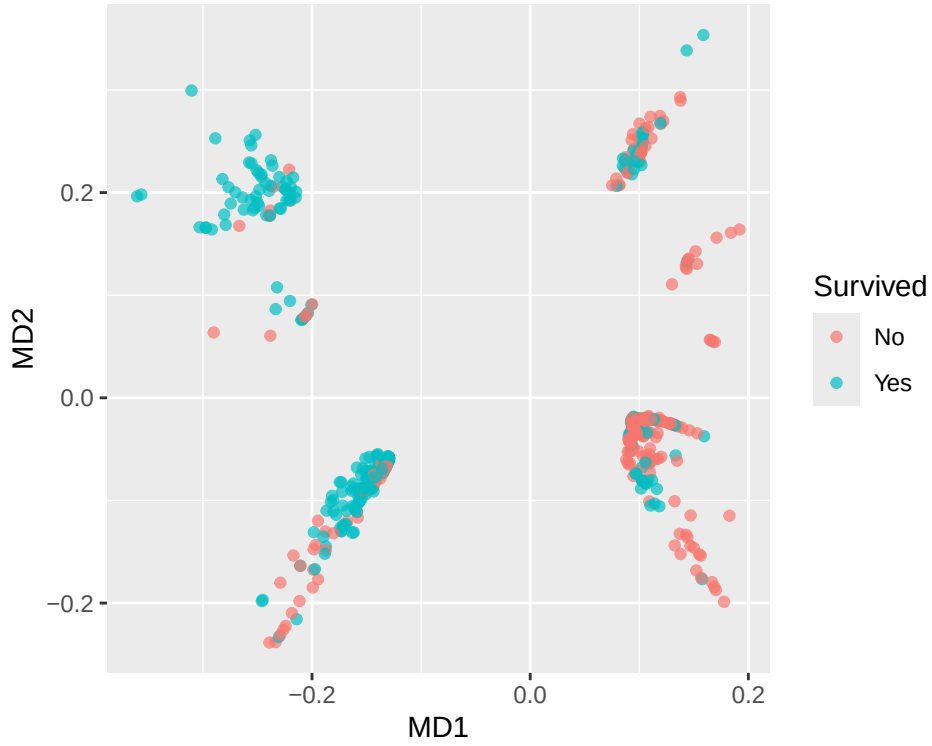


Miejscami dane znacząco odstają od prostej y = x, zatem jakość odwzorowania może nie być zadawalająca.

3.4 d.

Wizualizacja skalowania wielowymiarowego

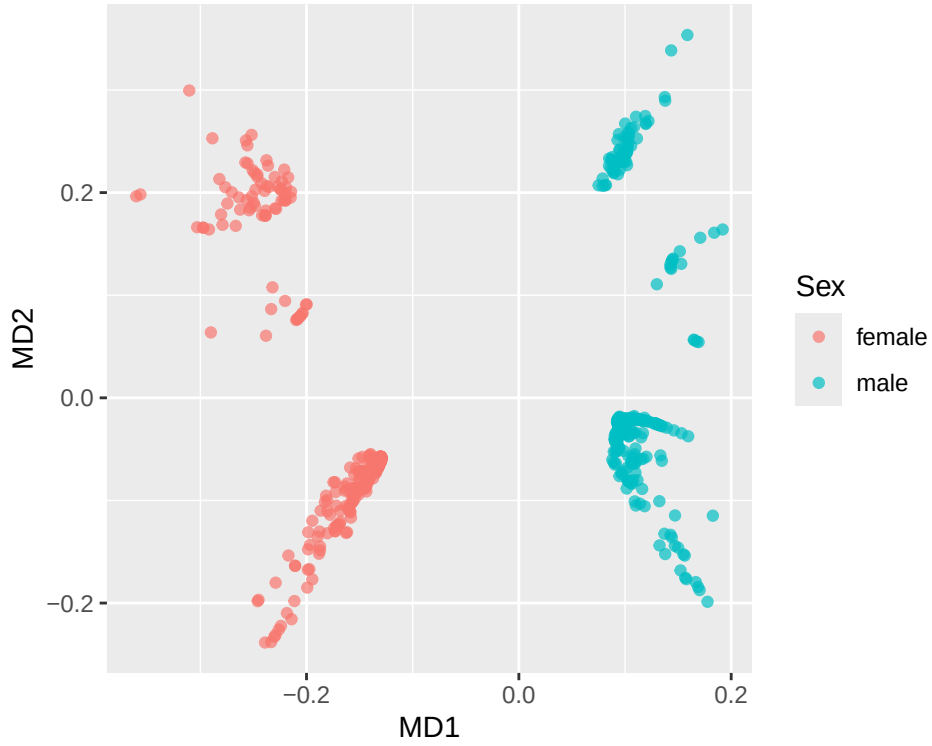
```r
data$MD1 <- mds.results[,1]
data$MD2 <- mds.results[,2]
data$Pclass <- Pclass

ggplot(data, aes(x = MD1, y = MD2, colour = Survived)) +
  geom_point(alpha = 0.7)
```



Widzimy, że dane układają się w sześć skupisk. Nie determinują one całkowicie przeżywalności, jednak możemy zauważyć, że przypadki o ujemnej wartości MD1 znacznie częściej uchodziły z życiem. Znajdujemy kilka nieznacznie odstających obserwacji w prawym górnym i w lewym górnym rogu. Mają one najwyższe wartości MD2 oraz ekstremalne wartości MD1.

Przeżywalność ze względu na płeć

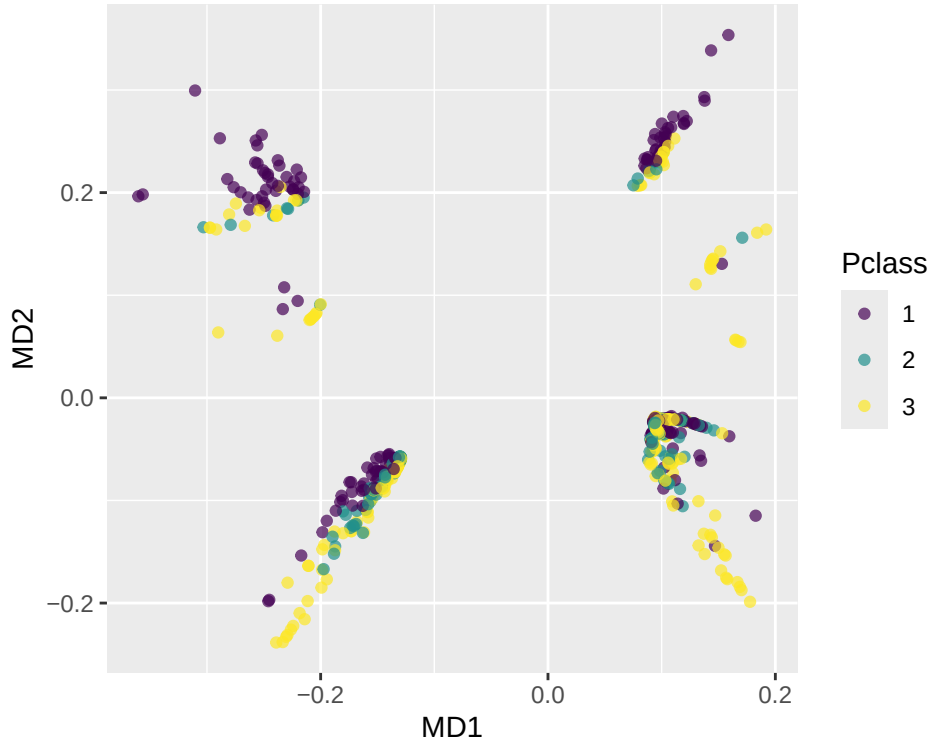
```
ggplot(data, aes(x = MD1, y = MD2, colour = Sex)) + geom_point(alpha = 0.7)
```



Teraz widzimy, że mężczyźni mają dodatnią wartość MD1, natomiast kobiety ujemną. Jest to najważnieszy czynnik determinujący przeżywalność.

Przeżywalność ze względu na klasę

```
ggplot(data, aes(x = MD1, y = MD2, colour = Pclass)) + geom_point(alpha = 0.7)
```




```
# procentwa przeżywalność w klasie 1
a <- nrow(data[data$Survived == "Yes" & data$Pclass == 1,])
b <- nrow(data[data$Pclass == 1,])
a/b
```

```
## [1] 0.655914
```

```
# procentwa przeżywalność w klasie 2
a <- nrow(data[data$Survived == "Yes" & data$Pclass == 2,])
b <- nrow(data[data$Pclass == 2,])
a/b
```

```
## [1] 0.4797688
```

```
# procentwa przeżywalność w klasie3
a <- nrow(data[data$Survived == "Yes" & data$Pclass == 3,])
b <- nrow(data[data$Pclass == 3,])
a/b
```

```
## [1] 0.2394366
```

Widzimy, że pasażerowie z lepszych klas mieli znacząco wyższą przeżywalność. Dają się zauważyć pewne skupiska w podziale na klasy, jendak nie są one wyraźne.