Kolokwium

Piotr Chlebicki

2024-01-29

Pakiety:

```
library(tidyverse) # dplyr + ggplot
library(TSA)
library(forecast)
library(caret)
```

Zadanie 1

a) Proces zadany przez:

$$Y_t = \frac{1}{3}Y_{t-1} + \frac{2}{9}Y_{t-2} + \varepsilon_t, t \geq 3$$

gdzie $\varepsilon_1, \ldots \sim \mathcal{N}(0,1)$ i.i.d. to process AR(2)

b) Wielomian charakterystyczny:

$$1 - \frac{1}{3}t - \frac{2}{9}t^2$$

moduły jego pierwiastków to:

```
polyroot(c(1, -1/3, -2/9)) |> abs()
```

```
## [1] 1.5 3.0
```

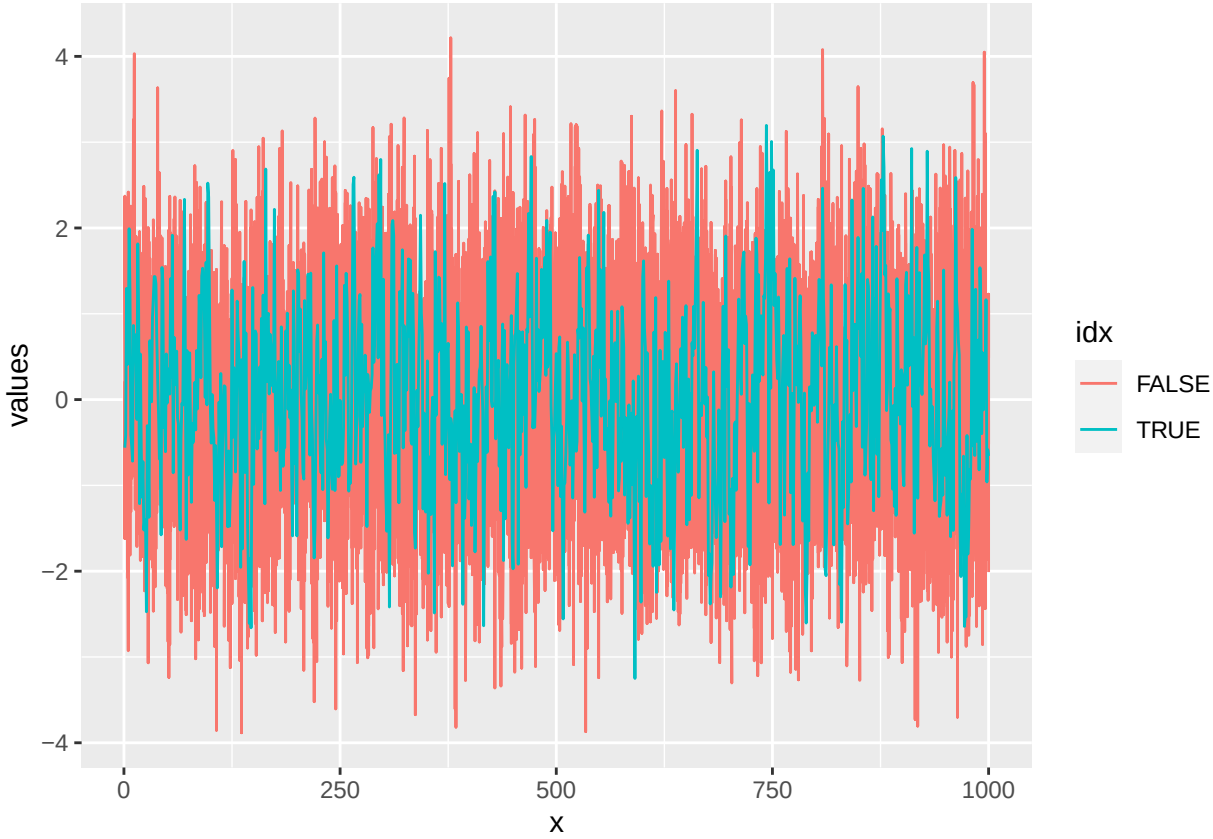
są one wszystkie większe (ostro) od 1 więc proces jest stacjonarny.

c) Generacja przy założeniu, że $Y_0 = \varepsilon_0$ i $Y_1 = \frac{1}{3}Y_0 + \varepsilon_1$

```
set.seed(123)
generate_y <- function(n = 1000L) {
  epsilon <- rnorm(n = n)
  Y <- vector(mode = "numeric", length = n)
  coef <- c(1/3, 2/9)
  Y[1] <- epsilon[1]
  Y[2] <- coef[1] * Y[1] + epsilon[2]
  k <- 3
  while (k <= n) {
    Y[k] <- t(coef) %*% c(Y[k - 1], Y[k - 2]) + epsilon[k]
    k <- k+1
  }
  Y
}
Y <- generate_y()
```

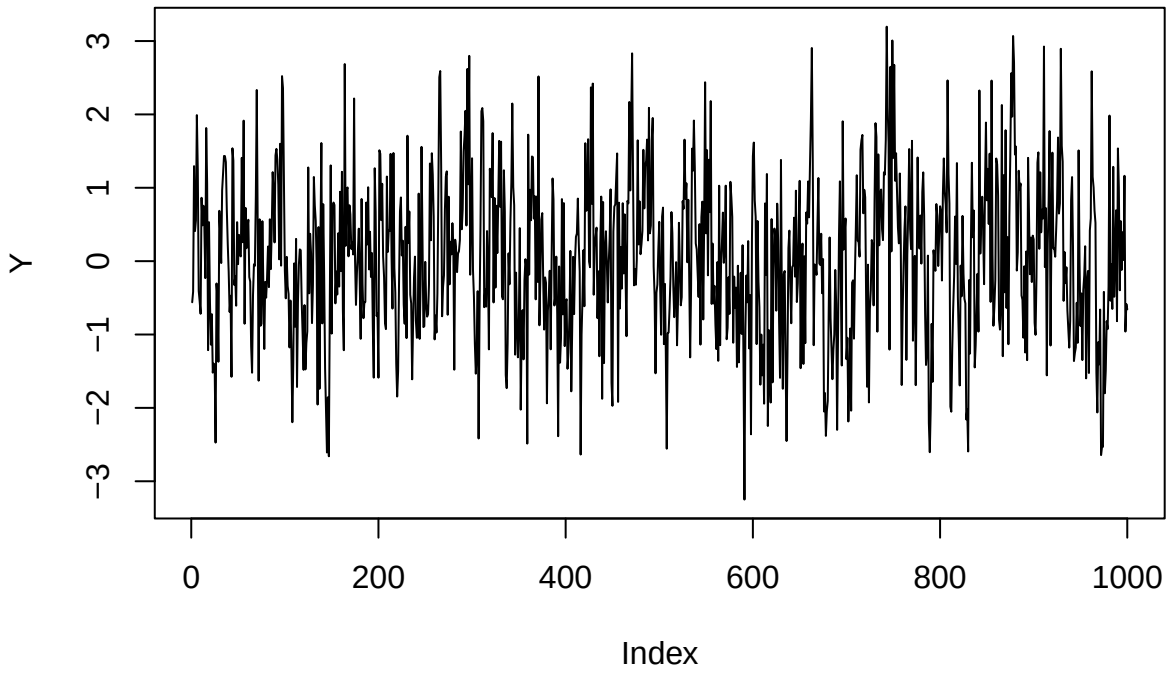
d) Wykres z kilkoma realizacjami niebieski to oryginalna trajektoria.

```
df <- do.call(rbind, lapply(1:10, FUN = function(x) generate_y())) |>
  t() |>
  cbind(Y) |>
  as.vector() |>
  data.frame(idx = (rep(1:11, each = 1000L) == 11) |> factor(),
             x = 1:1000L)

colnames(df)[1] <- "values"

df |>
  ggplot(aes(x = x, y = values, group = idx, col = idx)) +
  geom_line()
```



Wykres tylko wygynerowanej trajektorii:

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



Zad 2

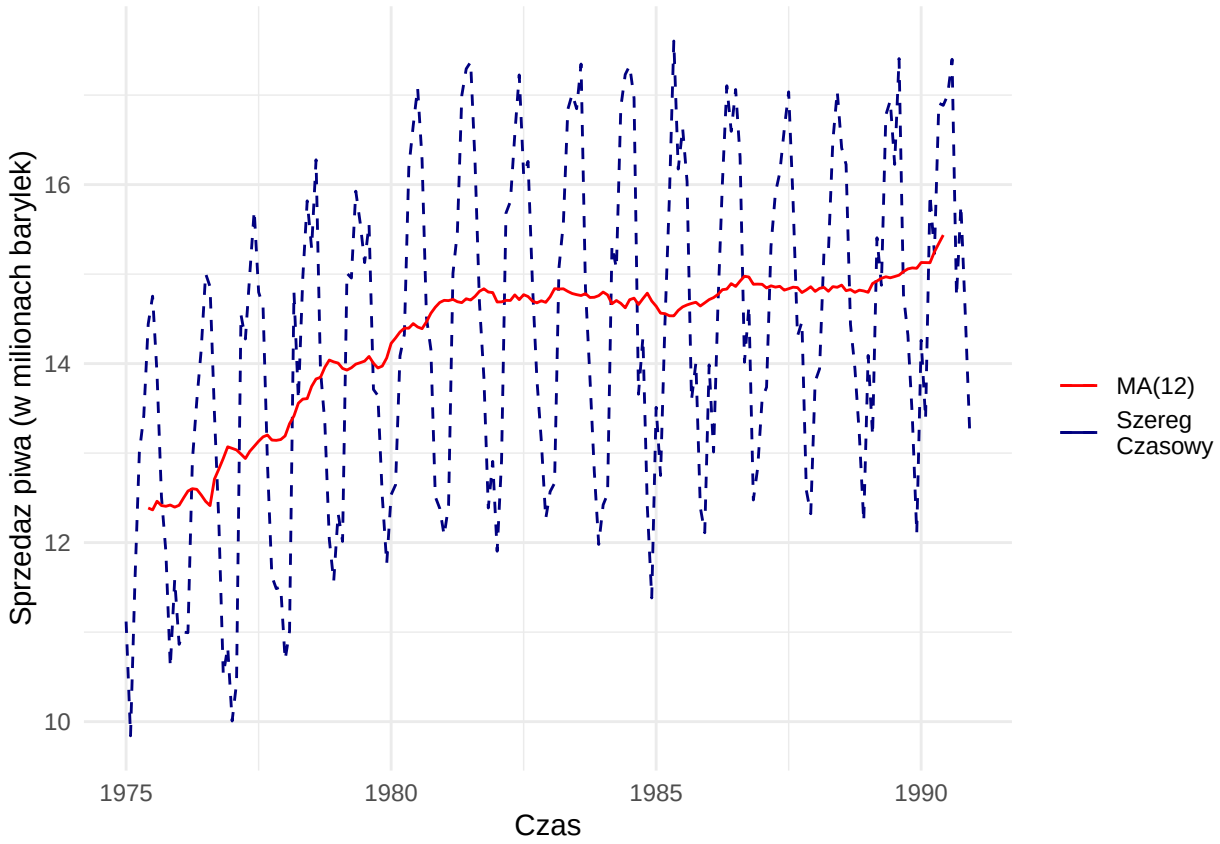
a) Wartość 12 została wybrana żeby wyeliminować potencjalne efekty sezonowe występujące w czasie roku

```
data("beersales")
df_aux <- data.frame(
  value = stats::filter(beersales, rep(1, 12) / 12) |> as.numeric(),
  time = stats::filter(beersales, rep(1, 12) / 12) |> time()
)
df <- tibble(
  value = beersales |> as.numeric(),
  time = beersales |> time()
)
```

```
df |>
  ggplot(aes(x = time, y = value)) +
  geom_line(lty = 2, aes(col = "Szereg\nCzasowy")) +
  geom_line(data = df_aux, aes(col = "MA(12)")) +
  scale_colour_manual(
    values = c(
      "Szereg\nCzasowy" = "navy",
      "MA(12)" = "red",
      "Arima" = "darkgreen"
    )
  ) +
  labs(x = "Czas",
       y = "Sprzedaż piwa (w milionach baryłek)",
       colour = "") +
  theme_minimal()
```

```
## Don't know how to automatically pick scale for object of type <ts>. Defaulting
## to continuous.
```

```
## Warning: Removed 11 rows containing missing values (`geom_line()`).
```



b) Na podstawie modelu MA(12) istnieje trend bo czerwona linia na wykresie (modelu) nie jest prostą stale równą średniej ogólnej. Występuje także sezonowość ponieważ zaobserwowane wartości w miesiącach letnich (zimowych) wyraźnie są wyższe (niższe) niż prognoza modelu.

c) Nie ma podstawy przypuszczać, że szereg jest stacjonarny (wysoka P-wartość).

```
tseries::adf.test(beersales, k = 12)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  beersales
## Dickey-Fuller = -1.8232, Lag order = 12, p-value = 0.6501
## alternative hypothesis: stationary
```

```
model <- auto.arima(beersales)
```

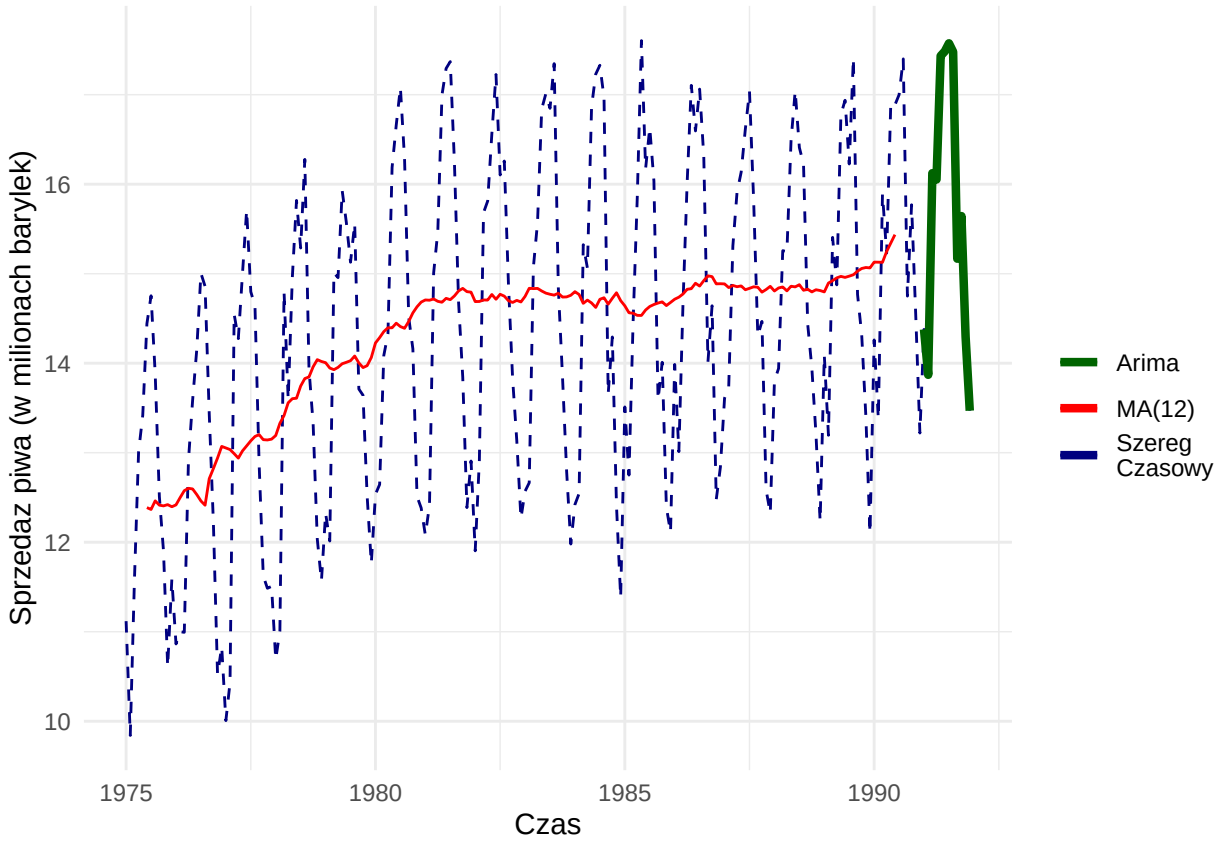
d)

```
df <- tibble(
  value = c(beersales, forecast(model, 12)$mean) |> as.numeric(),
  time = c(beersales |> time(), forecast(model, 12)$mean |> time())
)

df[, "iddx"] <- df$time < 1991
```

```r
df |>
  ggplot(aes(x = time, y = value)) +
  geom_line(lty = 2, aes(col = "Szereg\nCzasowy")) +
  geom_line(data = data.frame(
    value = forecast(model, 12)$mean |> as.numeric(),
    time = forecast(model, 12)$mean |> time()
  ), aes(col = "Arima"), linewidth = 1.5) +
  geom_line(data = df_aux, aes(col = "MA(12)")) +
  scale_colour_manual(
    values = c(
      "Szereg\nCzasowy" = "navy",
      "MA(12)" = "red",
      "Arima" = "darkgreen"
    )
  ) +
  labs(x = "Czas",
       y = "Sprzedaż piwa (w milionach baryłek)",
       colour = "") +
  theme_minimal()
```

e)

```
## Warning: Removed 11 rows containing missing values (`geom_line()`).
```



f) Prognozowana wartość sprzedaży piwa to

```r
forecast(model, 13)$mean[13]
```

```
## [1] 14.53585
```

Zadanie 3

a) Skalowanie jest potrzebne ponieważ wszystkie wartości reprezentują energie w różnych częstotliwościach są więc nieporównywalne.

```r
data(Sonar, package = "mlbench")
```

```r
model_pca <- prcomp(Sonar[, 1:60] |> scale())
```

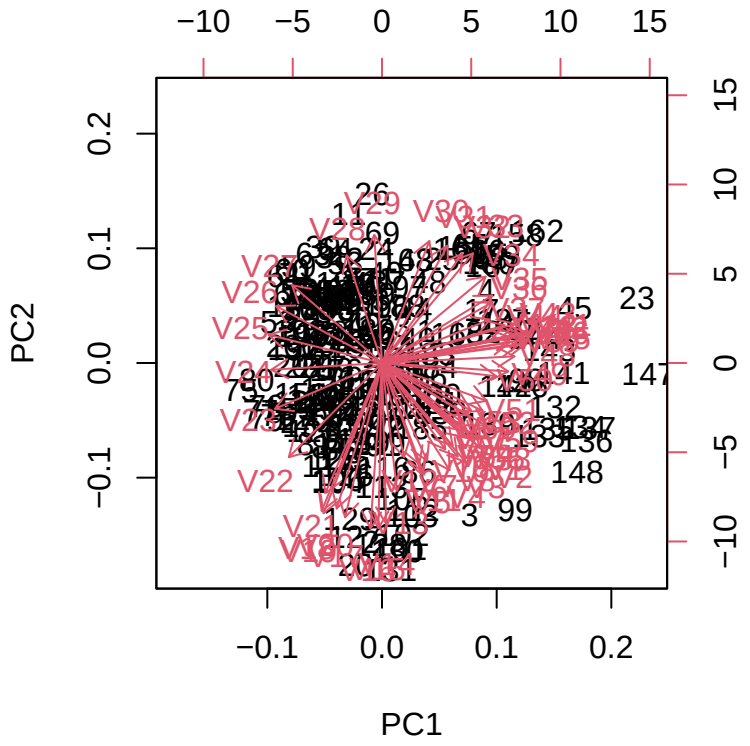
b) Pierwsze 2 składowe główne wyjaśniają 39.24% wariancji trzecia wyjaśnia 8.55% a powstało 60 składowych (bo tyle było zmiennych numerycznych)

```r
summary(model_pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.4940 3.3672 2.2649 1.84595 1.73328 1.56173 1.40264
## Proportion of Variance 0.2035 0.1890 0.0855 0.05679 0.05007 0.04065 0.03279
## Cumulative Proportion  0.2035 0.3924 0.4779 0.53473 0.58480 0.62545 0.65824
##                            PC8     PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation     1.35199 1.24080 1.22256 1.11587 1.06827 1.02381 0.96078
## Proportion of Variance 0.03046 0.02566 0.02491 0.02075 0.01902 0.01747 0.01538
## Cumulative Proportion  0.68870 0.71436 0.73928 0.76003 0.77905 0.79652 0.81190
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.92557 0.90365 0.86068 0.83737 0.78643 0.76642 0.75263
## Proportion of Variance 0.01428 0.01361 0.01235 0.01169 0.01031 0.00979 0.00944
## Cumulative Proportion  0.82618 0.83979 0.85214 0.86382 0.87413 0.88392 0.89336
##                           PC22   PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.72975 0.7099 0.68008 0.65815 0.64639 0.60764 0.56477
## Proportion of Variance 0.00888 0.0084 0.00771 0.00722 0.00696 0.00615 0.00532
## Cumulative Proportion  0.90224 0.9106 0.91834 0.92556 0.93253 0.93868 0.94400
##                           PC29    PC30    PC31    PC32    PC33    PC34    PC35
## Standard deviation     0.56110 0.54520 0.53453 0.51157 0.47273 0.44947 0.43163
## Proportion of Variance 0.00525 0.00495 0.00476 0.00436 0.00372 0.00337 0.00311
## Cumulative Proportion  0.94924 0.95420 0.95896 0.96332 0.96705 0.97041 0.97352
##                           PC36    PC37    PC38    PC39   PC40    PC41    PC42
## Standard deviation     0.42650 0.41547 0.38307 0.36290 0.3551 0.33277 0.30813
## Proportion of Variance 0.00303 0.00288 0.00245 0.00219 0.0021 0.00185 0.00158
## Cumulative Proportion  0.97655 0.97943 0.98187 0.98407 0.9862 0.98801 0.98960
##                           PC43    PC44    PC45    PC46    PC47    PC48    PC49
## Standard deviation     0.28645 0.27466 0.24827 0.23864 0.23726 0.20606 0.18873
## Proportion of Variance 0.00137 0.00126 0.00103 0.00095 0.00094 0.00071 0.00059
## Cumulative Proportion  0.99096 0.99222 0.99325 0.99420 0.99514 0.99584 0.99644
##                           PC50    PC51    PC52    PC53    PC54    PC55    PC56
## Standard deviation     0.17686 0.17099 0.16868 0.15174 0.14833 0.13897 0.12733
## Proportion of Variance 0.00052 0.00049 0.00047 0.00038 0.00037 0.00032 0.00027
## Cumulative Proportion  0.99696 0.99745 0.99792 0.99830 0.99867 0.99899 0.99926
##                           PC57    PC58    PC59    PC60
## Standard deviation     0.12160 0.10768 0.10597 0.08128
## Proportion of Variance 0.00025 0.00019 0.00019 0.00011
## Cumulative Proportion  0.99951 0.99970 0.99989 1.00000
```

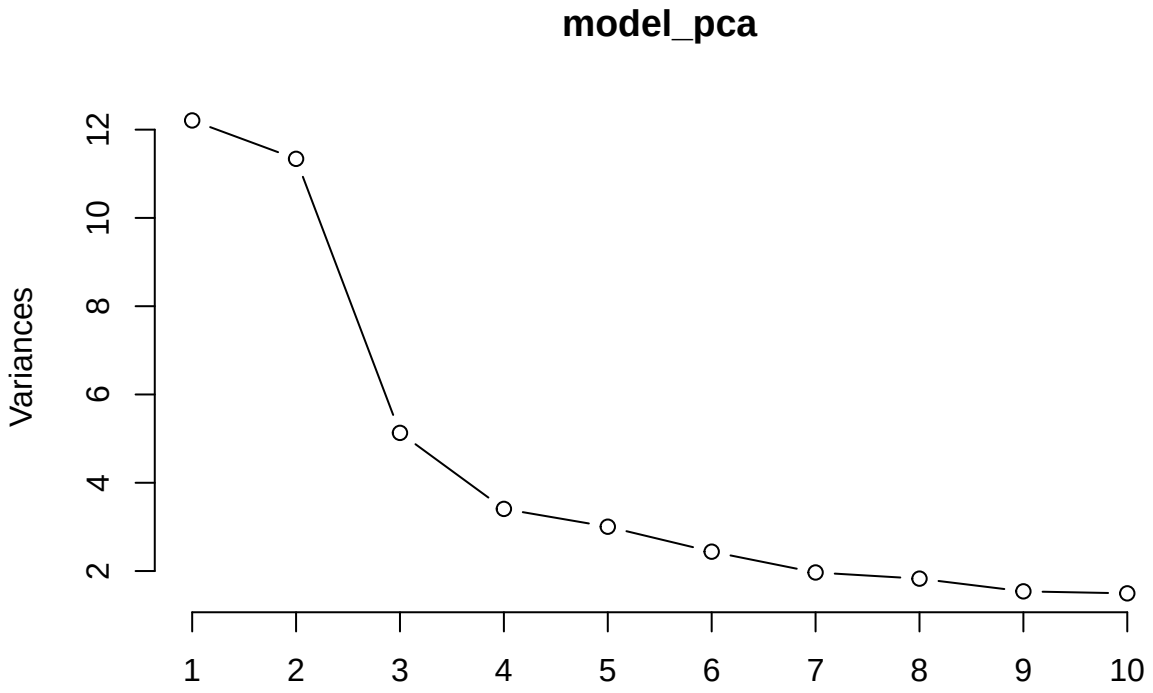
c) Biplot:

```
biplot(model_pca)
```



Wykres osypiska

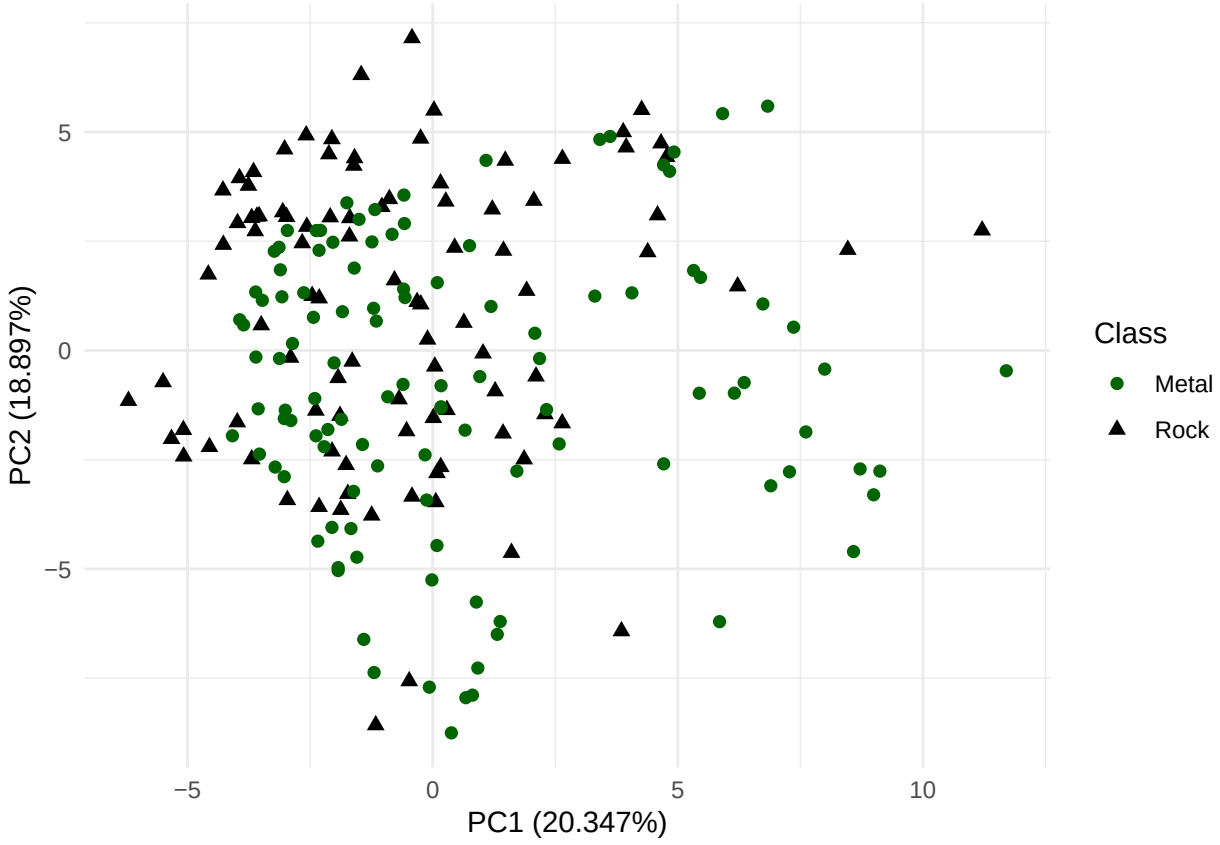
```
plot(model_pca, type = "lines")
```



d) Dwie pierwsze składowe główne nie wyjaśniają niestety nawet połowę zmienności w danych więc zostawienie tylko dwóch składowych jest niewłaściwe.

```r
df <- cbind(model_pca$x, Class = Sonar$Class) |> as_tibble()
df$Class <- (df$Class == 2) |> ifelse("Rock", "Metal") |> factor()
```

```r
df |>
  ggplot(aes(x = PC1, y = PC2, shape = Class, col = Class)) +
  geom_point(size = 2) +
  scale_colour_manual(
    values = c(
      "Rock" = "black",
      "Metal" = "darkgreen"
    )
  ) +
  labs(x = paste0("PC1", " (", summary(model_pca)$importance[2, 1] * 100, "%)"),
       y = paste0("PC2", " (", summary(model_pca)$importance[2, 2] * 100, "%)"),
       colour = "Class") +
  theme_minimal()
```



e)

```r
model_rf <- train(
  Class ~ PC1 + PC2 + PC3 + PC4 + PC5,
  data = df,
  method = "ranger",
  trControl = trainControl(method = "cv", number = 10)
)
```

```
confusionMatrix(model_rf)
```

f)

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction Metal Rock
##      Metal  44.7  8.7
##      Rock    8.7 38.0
##
##  Accuracy (average) : 0.8269
```

```
confusionMatrix(predict(model_rf), df$Class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Metal Rock
##      Metal   111    0
##      Rock      0   97
##
##                Accuracy : 1
##                  95% CI : (0.9824, 1)
##     No Information Rate : 0.5337
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5337
##          Detection Rate : 0.5337
##    Detection Prevalence : 0.5337
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : Metal
##
```

Model różni się istotnie od losowego modelu kórego dokładność byłaby na poziomie około 53%.

g) Obserwacja została zaklasifikowana do kamieni.

```
set.seed(999)
ddf <- runif(60) |> data.frame() |> t()

colnames(ddf) <- paste0("PC", 1:60)
predict(model_rf, newdata = ddf)
```

```
## [1] Rock
## Levels: Metal Rock
```