lista10

Lista 10

a) Utwórz zbiory: treningowy zawierający 80% danych oraz testowy zawierający 20% danych.

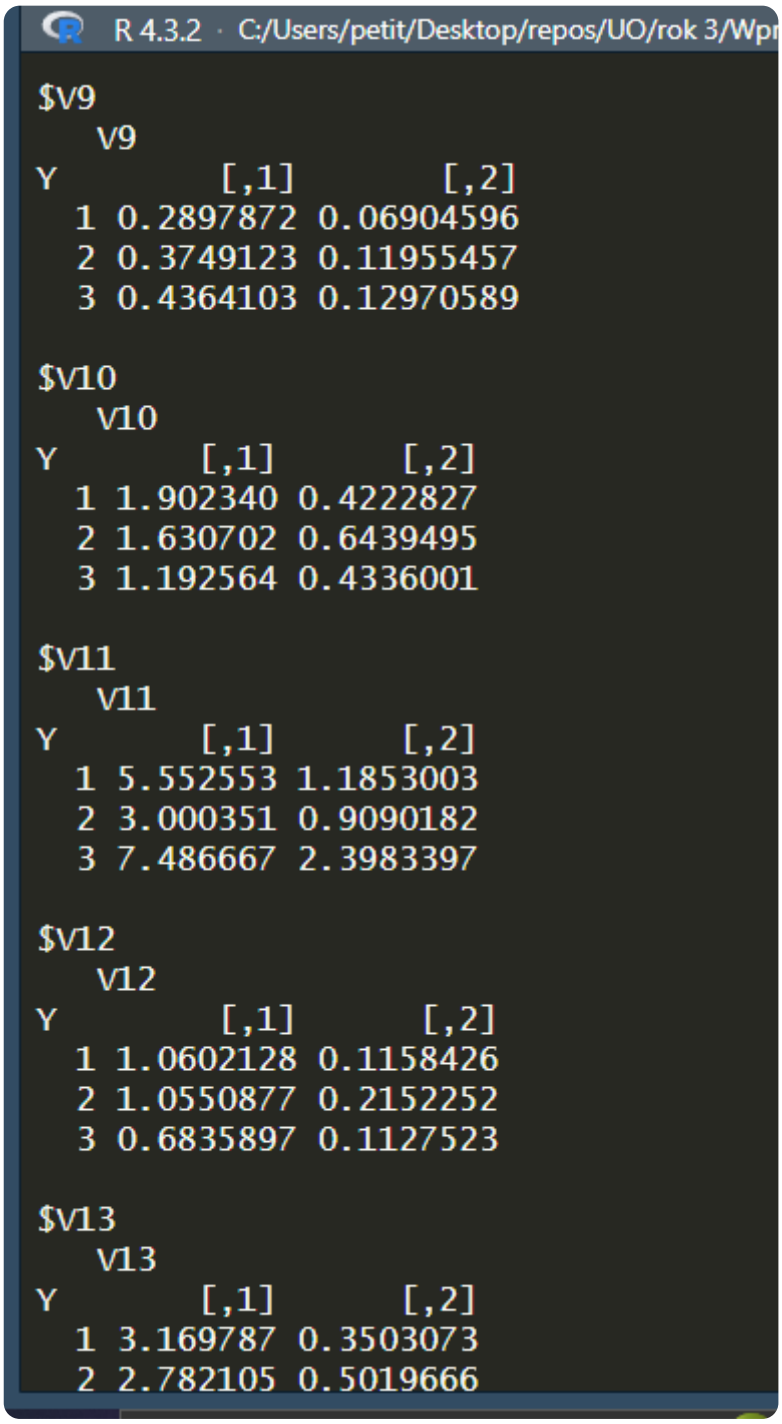
```r
# Ustawienie ziarna losowości dla powtarzalności wyników
set.seed(123)

# Podział danych na zbiór treningowy i testowy
splitIndex <- createDataPartition(wine_data$V1, p = 0.8, list = FALSE)
train_set <- wine_data[splitIndex, ]
test_set <- wine_data[-splitIndex, ]
```

b) Zbuduj naiwny klasyfikator bayesowski (na zbiorze treningowym). Wyświetl rozkład klas dla zmiennej celu oraz listę tabel prawdopodobieństw warunkowych.

```r
# Budowanie naiwnego klasyfikatora bayesowskiego
nb_model <- naiveBayes(V1 ~ ., data = train_set)

# Wyświetlenie rozkładu klas zmiennej celu
print(nb_model$class)

# Wyświetlenie tabel prawdopodobieństw warunkowych dla każdej zmiennej
print(nb_model$tables)
```

```
>
> # Wyświetlenie tabel prawdopodobieństw warunkowych dla każdej zmiennej
> print(nb_model$tables)
$V2
   V2
Y       [,1]      [,2]
  1 13.78702 0.4583473
  2 12.24895 0.5278145
  3 13.20974 0.5384895

$V3
   V3
Y       [,1]      [,2]
  1 2.067021 0.7476948
  2 2.021053 1.0957298
  3 3.278718 1.1461358

$V4
   V4
Y       [,1]      [,2]
  1 2.467234 0.2294388
  2 2.246140 0.3131166
  3 2.436923 0.1941805
```

```
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/ro
  1 2.467234 0.2294388
  2 2.246140 0.3131166
  3 2.436923 0.1941805

$V5
   V5
Y       [,1]     [,2]
  1 16.95106 2.558691
  2 20.27018 3.455222
  3 21.39744 2.206985

$V6
   V6
Y        [,1]     [,2]
  1 105.89362 10.75262
  2  94.96491 17.53361
  3  99.87179 10.55108

$V7
   V7
Y       [,1]      [,2]
  1 2.839574 0.3468990
  2 2.290526 0.4994441
  3 1.672051 0.3451939

$V8
   V8
Y        [,1]      [,2]
  1 2.9808511 0.3681377
  2 2.0922807 0.7101610
  3 0.8148718 0.3093190

$V9
   V9
```


R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wpr
$V9
V9
Y [,1] [,2]
1 0.2897872 0.06904596
2 0.3749123 0.11955457
3 0.4364103 0.12970589
$V10
V10
Y [,1] [,2]
1 1.902340 0.4222827
2 1.630702 0.6439495
3 1.192564 0.4336001
$V11
V11
Y [,1] [,2]
1 5.552553 1.1853003
2 3.000351 0.9090182
3 7.486667 2.3983397
$V12
V12
Y [,1] [,2]
1 1.0602128 0.1158426
2 1.0550877 0.2152252
3 0.6835897 0.1127523
$V13
V13
Y [,1] [,2]
1 3.169787 0.3503073
2 2.782105 0.5019666

```
$V13
   V13
Y       [,1]      [,2]
  1 3.169787 0.3503073
  2 2.782105 0.5019666
  3 1.664872 0.2755461

$V14
   V14
Y        [,1]     [,2]
  1 1108.0213 212.6916
  2  514.0702 158.9408
  3  626.5385 108.4530

>
```

c) Przeprowadź klasyfikację danych metodą naiwnego klasyfikatora bayesowskiego (na zbiorze testowym). Wyświetl macierz błędów.

```r
# Przewidywanie na zbiorze testowym
predictions <- predict(nb_model, test_set)

# Obliczenie i wyświetlenie macierzy błędów
confusionMatrix <- table(test_set$V1, predictions)
print(confusionMatrix)
```

```
>
> # Obliczenie i wyświetlenie macierzy błędów
> confusionMatrix <- table(test_set$V1, predictions)
> print(confusionMatrix)
   predictions
     1  2  3
  1 12  0  0
  2  0 13  1
  3  0  0  9
> # Załadowanie potrzebnej biblioteki
> library(caret)
```

d) Napisz, jaka jest dokładność klasyfikacji i współczynnik błędu

Na podstawie macierzy błędów można obliczyć dokładność klasyfikacji oraz współczynnik błędu. Dokładność klasyfikacji to stosunek liczby poprawnie sklasyfikowanych przypadków do całkowitej liczby przypadków, a

współczynnik błędu to stosunek liczby błędnie sklasyfikowanych przypadków do całkowitej liczby przypadków.

Macierz błędów wygląda następująco:

```
   predictions
     1  2  3
1 12  0  0
2  0 13  1
3  0  0  9
```

Obliczmy teraz dokładność i współczynnik błędu:

1. **Dokładność (Accuracy)**: Jest to `(poprawne przypisania klasy 1 + poprawne przypisania klasy 2 + poprawne przypisania klasy 3) / całkowita liczba przypadków`
   - W naszym przypadku: `(12 + 13 + 9) / (12 + 13 + 1 + 9)`
2. **Współczynnik błędu (Error Rate)**: Jest to `1 - dokładność`
   - Można to również obliczyć jako `(błędne przypisania) / całkowita liczba przypadków`

Obliczymy te wartości:

Dokładność klasyfikacji wynosi około (97.14%) a współczynnik błędu to około (2.86%). Oznacza to, że model naiwnego klasyfikatora bayesowskiego bardzo dobrze radzi sobie z klasyfikacją na zbiorze testowym.

e) Zbuduj cztery modele SVM z różnymi funkcjami jądra: radialną, liniową, wielomianową i sigmoidalną (pamiętaj o standaryzacji danych).

```r
# Standaryzacja danych treningowych i testowych
train_set_standardized <- scale(train_set[, -1])
test_set_standardized <- scale(test_set[, -1], center = attr(train_set_standardized, "scaled:center"), s

# Dodanie kolumny z etykietami klasy do zbiorów danych
train_set_standardized <- data.frame(train_set$V1, train_set_standardized)
test_set_standardized <- data.frame(test_set$V1, test_set_standardized)
colnames(train_set_standardized)[1] <- "V1"
colnames(test_set_standardized)[1] <- "V1"

# Budowanie modeli SVM z różnymi funkcjami jądra
svm_radial <- svm(V1 ~ ., data = train_set_standardized, kernel = "radial")
svm_linear <- svm(V1 ~ ., data = train_set_standardized, kernel = "linear")
svm_polynomial <- svm(V1 ~ ., data = train_set_standardized, kernel = "polynomial")
svm_sigmoid <- svm(V1 ~ ., data = train_set_standardized, kernel = "sigmoid")

# Wyświetlenie informacji o modelach
summary(svm_radial)
summary(svm_linear)
summary(svm_polynomial)
summary(svm_sigmoid)
```

```
> # Wyświetlenie informacji o modelach
> summary(svm_radial)

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "radial")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.07692308
    epsilon:  0.1


Number of Support Vectors:  84
```

```
> summary(svm_linear)

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "linear")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  linear
       cost:  1
      gamma:  0.07692308
    epsilon:  0.1


Number of Support Vectors:  107
```

```
> summary(svm_polynomial)

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "polynomial")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  polynomial
       cost:  1
     degree:  3
      gamma:  0.07692308
     coef.0:  0
    epsilon:  0.1


Number of Support Vectors:  112
```

```
> summary(svm_sigmoid)

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "sigmoid")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  sigmoid
       cost:  1
      gamma:  0.07692308
     coef.0:  0
    epsilon:  0.1


Number of Support Vectors:  134
```

f) Dla każdego modelu wydrukuj parametry i liczbę wektorów nośnych

```r
# Wyświetlenie parametrów i liczby wektorów nośnych dla każdego modelu SVM

# Model SVM z funkcją jądra radialną
cat("Model SVM z funkcją jądra radialną:\n")
print(summary(svm_radial))
cat("Liczba wektorów nośnych:", length(svm_radial$index), "\n\n")

# Model SVM z funkcją jądra liniową
cat("Model SVM z funkcją jądra liniową:\n")
print(summary(svm_linear))
cat("Liczba wektorów nośnych:", length(svm_linear$index), "\n\n")

# Model SVM z funkcją jądra wielomianową
cat("Model SVM z funkcją jądra wielomianową:\n")
print(summary(svm_polynomial))
cat("Liczba wektorów nośnych:", length(svm_polynomial$index), "\n\n")

# Model SVM z funkcją jądra sigmoidalną
cat("Model SVM z funkcją jądra sigmoidalną:\n")
print(summary(svm_sigmoid))
cat("Liczba wektorów nośnych:", length(svm_sigmoid$index), "\n\n")
```

```
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wprowadzenie do eksploracji danych/lista12/
> # Wyświetlenie parametrów i liczby wektorów nośnych dla każdego modelu SVM
>
> # Model SVM z funkcją jądra radialną
> cat("Model SVM z funkcją jądra radialną:\n")
Model SVM z funkcją jądra radialną:
> print(summary(svm_radial))

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "radial")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.07692308
    epsilon:  0.1


Number of Support Vectors:  84
```

```
> cat("Liczba wektorów nośnych:", length(svm_radial$index), "\n\n")
Liczba wektorów nośnych: 84

>
> # Model SVM z funkcją jądra liniową
> cat("Model SVM z funkcją jądra liniową:\n")
Model SVM z funkcją jądra liniową:
> print(summary(svm_linear))

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "linear")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  linear
       cost:  1
      gamma:  0.07692308
    epsilon:  0.1


Number of Support Vectors:  107
```

```
> cat("Liczba wektorów nośnych:", length(svm_linear$index), "\n\n")
Liczba wektorów nośnych: 107

>
> # Model SVM z funkcją jądra wielomianową
> cat("Model SVM z funkcją jądra wielomianową:\n")
Model SVM z funkcją jądra wielomianową:
> print(summary(svm_polynomial))

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "polynomial")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  polynomial
       cost:  1
     degree:  3
      gamma:  0.07692308
     coef.0:  0
    epsilon:  0.1


Number of Support Vectors:  112
```

```
> cat("Liczba wektorów nośnych:", length(svm_polynomial$index), "\n\n")
Liczba wektorów nośnych: 112

>
> # Model SVM z funkcją jądra sigmoidalną
> cat("Model SVM z funkcją jądra sigmoidalną:\n")
Model SVM z funkcją jądra sigmoidalną:
> print(summary(svm_sigmoid))

Call:
svm(formula = V1 ~ ., data = train_set_standardized, kernel = "sigmoid")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  sigmoid
       cost:  1
      gamma:  0.07692308
     coef.0:  0
    epsilon:  0.1


Number of Support Vectors:  134
```

```
> cat("Liczba wektorów nośnych:", length(svm_sigmoid$index), "\n\n")
Liczba wektorów nośnych: 134

>
```

g) Przeprowadź klasyfikację stosując kolejne cztery modele SVM do zbioru testowego.

```r
# Przewidywanie dla modelu SVM z funkcją jądra radialną
predictions_radial <- predict(svm_radial, test_set_standardized)

# Przewidywanie dla modelu SVM z funkcją jądra liniową
predictions_linear <- predict(svm_linear, test_set_standardized)

# Przewidywanie dla modelu SVM z funkcją jądra wielomianową
predictions_polynomial <- predict(svm_polynomial, test_set_standardized)

# Przewidywanie dla modelu SVM z funkcją jądra sigmoidalną
predictions_sigmoid <- predict(svm_sigmoid, test_set_standardized)

# Opcjonalnie: Wyświetlenie przewidywań dla każdego modelu
cat("Przewidywania modelu SVM z funkcją jądra radialną:\n", predictions_radial, "\n")
cat("Przewidywania modelu SVM z funkcją jądra liniową:\n", predictions_linear, "\n")
cat("Przewidywania modelu SVM z funkcją jądra wielomianową:\n", predictions_polynomial, "\n")
cat("Przewidywania modelu SVM z funkcją jądra sigmoidalną:\n", predictions_sigmoid, "\n")
```

```
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wprowadzenie do eksploracji danych/lista12/
> # g) Przeprowadź klasyfikację stosując kolejne cztery modele SVM do zbioru testowego.
> # Przewidywanie dla modelu SVM z funkcją jądra radialną
> predictions_radial <- predict(svm_radial, test_set_standardized)
>
> # Przewidywanie dla modelu SVM z funkcją jądra liniową
> predictions_linear <- predict(svm_linear, test_set_standardized)
>
> # Przewidywanie dla modelu SVM z funkcją jądra wielomianową
> predictions_polynomial <- predict(svm_polynomial, test_set_standardized)
>
> # Przewidywanie dla modelu SVM z funkcją jądra sigmoidalną
> predictions_sigmoid <- predict(svm_sigmoid, test_set_standardized)
>
> # Opcjonalnie: Wyświetlenie przewidywań dla każdego modelu
> cat("Przewidywania modelu SVM z funkcją jądra radialną:\n", predictions_radial, "\n")
Przewidywania modelu SVM z funkcją jądra radialną:
 0.9253141 1.154281 1.116268 0.9513025 1.014082 1.273846 1.161947 1.217867 0.9674315 1.197292 1.060859 1.013441 2.092303
1.563645 1.844207 2.470899 2.195967 2.018954 1.964491 2.091244 2.341307 2.098257 1.905313 1.980307 2.075071 1.807855 2.76
1411 2.755826 3.040487 2.789695 2.861382 2.910758 3.091882 2.986817 2.994976
> cat("Przewidywania modelu SVM z funkcją jądra liniową:\n", predictions_linear, "\n")
Przewidywania modelu SVM z funkcją jądra liniową:
 0.8094901 0.9331381 1.540173 0.7950694 1.085017 0.4979976 1.264076 1.376328 1.159678 1.302604 1.492177 0.9374362 1.81126
9 1.682841 1.475441 2.337042 2.19202 2.061338 2.125274 2.108927 2.556291 2.119183 1.796703 1.917395 2.146846 1.481126 2.8
58266 3.148599 3.031552 3.009251 3.11184 2.726811 2.693352 3.043791 2.834147
> cat("Przewidywania modelu SVM z funkcją jądra wielomianową:\n", predictions_polynomial, "\n")
Przewidywania modelu SVM z funkcją jądra wielomianową:
 0.7959544 1.212769 1.605373 0.5229623 1.312196 -0.1003526 1.547861 1.578948 1.622152 1.440796 1.561803 1.229071 1.721158
1.751412 1.315444 2.086959 1.843153 2.257362 1.919908 1.928508 2.38733 1.991347 1.866106 1.957231 1.85431 1.490032 2.3355
74 4.66244 3.058929 3.15406 2.819389 2.52278 2.661924 2.840083 2.610935
> cat("Przewidywania modelu SVM z funkcją jądra sigmoidalną:\n", predictions_sigmoid, "\n")
Przewidywania modelu SVM z funkcją jądra sigmoidalną:
 1.391144 0.6549497 1.165122 1.601402 0.5623435 0.1709588 1.193469 1.448903 1.222446 1.460712 1.17915 1.021778 0.7971102
1.39472 0.2545922 2.938465 2.06291 1.399216 2.830091 1.293409 2.496483 2.479882 1.63427 3.036401 2.657165 1.218626 2.7784
17 1.505867 2.074757 2.733533 2.401592 2.381235 2.639565 2.36698 3.594083
>
```

h) Dla każdego modelu wydrukuj macierz błędów oraz dokładność.

```r
# Funkcja do obliczenia i wydrukowania macierzy błędów oraz dokładności
print_confusion_matrix_and_accuracy <- function(predictions, actual) {
  confusion_matrix <- table(actual, predictions)
  accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
  cat("Macierz błędów:\n")
  print(confusion_matrix)
  cat("Dokładność:", accuracy, "\n\n")
}

# Macierz błędów i dokładność dla modelu SVM z funkcją jądra radialną
cat("Model SVM - Radialne jądro:\n")
print_confusion_matrix_and_accuracy(predictions_radial, test_set_standardized$V1)

# Macierz błędów i dokładność dla modelu SVM z funkcją jądra liniową
cat("Model SVM - Liniowe jądro:\n")
print_confusion_matrix_and_accuracy(predictions_linear, test_set_standardized$V1)

# Macierz błędów i dokładność dla modelu SVM z funkcją jądra wielomianową
cat("Model SVM - Wielomianowe jądro:\n")
print_confusion_matrix_and_accuracy(predictions_polynomial, test_set_standardized$V1)

# Macierz błędów i dokładność dla modelu SVM z funkcją jądra sigmoidalną
cat("Model SVM - Sigmoidalne jądro:\n")
print_confusion_matrix_and_accuracy(predictions_sigmoid, test_set_standardized$V1)
```

```
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wprowadzenie do eksploracji danych/lista12/
> # h) Dla każdego modelu wydrukuj macierz błędów oraz dokładność.
> # Funkcja do obliczenia i wydrukowania macierzy błędów oraz dokładności
> print_confusion_matrix_and_accuracy <- function(predictions, actual) {
+   confusion_matrix <- table(actual, predictions)
+   accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
+   cat("Macierz błędów:\n")
+   print(confusion_matrix)
+   cat("Dokładność:", accuracy, "\n\n")
+ }
>
> # Macierz błędów i dokładność dla modelu SVM z funkcją jądra radialną
> cat("Model SVM - Radialne jądro:\n")
Model SVM - Radialne jądro:
> print_confusion_matrix_and_accuracy(predictions_radial, test_set_standardized$V1)
Macierz błędów:
      predictions
actual 0.925314061917655 0.951302463591675 0.96743145550474 1.01344130443722 1.01408157518439 1.0608589829982
     1                 1                 1                1                1                1                1
     2                 0                 0                0                0                0                0
     3                 0                 0                0                0                0                0
      predictions
actual 1.11626807135943 1.15428105285456 1.1619472632833 1.19729208602257 1.21786676979851 1.27384556398085
     1                1                1               1                1                1                1
     2                0                0               0                0                0                0
     3                0                0               0                0                0                0
      predictions
actual 1.56364466459154 1.80785530184693 1.84420742657521 1.90531348150749 1.96449129738224 1.98030693579788
     1                0                0                0                0                0                0
     2                1                1                1                1                1                1
     3                0                0                0                0                0                0
      predictions
actual 2.01895400822711 2.07507105952655 2.09124389654516 2.09230255020784 2.09825707102956 2.19596726221885
     1                0                0                0                0                0                0
     2                1                1                1                1                1                1
     3                0                0                0                0                0                0
      predictions
actual 2.34130718226162 2.47089936787415 2.75582637147352 2.7614109322342 2.7896947717802 2.86138166055648
     1                0                0                0                0                0                0
     2                1                1                0                0                0                0
     3                0                0                1                1                1                1
      predictions
actual 2.91075817075212 2.98681695850260 3.00407566236205 3.04048600060770 3.09188187752436
```

```
Console  Terminal  Background Jobs
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wprowadzenie do eksploracji danych/lista12/
     3                 0                 0                 0                 0                 0                 0
      predictions
actual 1.11626807135943 1.15428105285456 1.1619472632833 1.19729208602257 1.21786676979851 1.27384556398085
     1                 1                 1                 1                 1                 1
     2                 0                 0                 0                 0                 0
     3                 0                 0                 0                 0                 0
      predictions
actual 1.56364466459154 1.80785530184693 1.84420742657521 1.90531348150749 1.96449129738224 1.98030693579788
     1                 0                 0                 0                 0                 0
     2                 1                 1                 1                 1                 1
     3                 0                 0                 0                 0                 0
      predictions
actual 2.01895400822711 2.07507105952655 2.09124389654516 2.09230255020784 2.09825707102956 2.19596726221885
     1                 0                 0                 0                 0                 0
     2                 1                 1                 1                 1                 1
     3                 0                 0                 0                 0                 0
      predictions
actual 2.34130718226162 2.47089936787415 2.75582637147352 2.7614109322342 2.7896947717802 2.86138166055648
     1                 0                 0                 0                 0                 0
     2                 1                 1                 0                 0                 0
     3                 0                 0                 1                 1                 1
      predictions
actual 2.91075817975212 2.98681695859269 2.99497566236295 3.04048690069779 3.09188187752436
     1                 0                 0                 0                 0                 0
     2                 0                 0                 0                 0                 0
     3                 1                 1                 1                 1                 1
Dokładność: 0.02857143

>
> # Macierz błędów i dokładność dla modelu SVM z funkcją jądra liniową
> cat("Model SVM - Liniowe jądro:\n")
Model SVM - Liniowe jądro:
> print_confusion_matrix_and_accuracy(predictions_linear, test_set_standardized$V1)
Macierz błędów:
      predictions
actual 0.49799764361693 0.795069388465165 0.809490130475751 0.933138093878224 0.937436168452605 1.0850169541715
     1                 1                 1                 1                 1                 1
     2                 0                 0                 0                 0                 0                 0
     3                 0                 0                 0                 0                 0                 0
      predictions
actual 1.15967791577823 1.26407627341993 1.30260415382717 1.37632810770321 1.47544064048216 1.48112621470047
```

```
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wprowadzenie do eksploracji danych/lista12/
>
> # Macierz błędów i dokładność dla modelu SVM z funkcją jądra liniową
> cat("Model SVM - Liniowe jądro:\n")
Model SVM - Liniowe jądro:
> print_confusion_matrix_and_accuracy(predictions_linear, test_set_standardized$V1)
Macierz błędów:
      predictions
actual 0.49799764361693 0.795069388465165 0.809490130475751 0.933138093878224 0.937436168452605 1.0850169541715
     1                 1                 1                 1                 1                 1                 1
     2                 0                 0                 0                 0                 0                 0
     3                 0                 0                 0                 0                 0                 0
      predictions
actual 1.15967791577823 1.26407627341993 1.30260415382717 1.37632810770321 1.47544064048216 1.48112621470047
     1                 1                 1                 1                 1                 0                 0
     2                 0                 0                 0                 0                 1                 1
     3                 0                 0                 0                 0                 0                 0
      predictions
actual 1.49217719560556 1.54017342467057 1.68284134547141 1.79670329278938 1.81126901629309 1.91739528566033
     1                 1                 1                 0                 0                 0                 0
     2                 0                 0                 1                 1                 1                 1
     3                 0                 0                 0                 0                 0                 0
      predictions
actual 2.06133752237619 2.10892712485623 2.11918311301058 2.12527376734032 2.14684602288568 2.19201995082862
     1                 0                 0                 0                 0                 0                 0
     2                 1                 1                 1                 1                 1                 1
     3                 0                 0                 0                 0                 0                 0
      predictions
actual 2.33704193038442 2.55629087695542 2.69335178463739 2.72681149474552 2.83414677761992 2.85826646629042
     1                 0                 0                 0                 0                 0                 0
     2                 1                 1                 0                 0                 0                 0
     3                 0                 0                 1                 1                 1                 1
      predictions
actual 3.00925057736632 3.03155195050385 3.04379132580285 3.11184040364247 3.14859914313101
     1                 0                 0                 0                 0                 0
     2                 0                 0                 0                 0                 0
     3                 1                 1                 1                 1                 1
Dokładność: 0.02857143

>
```

```
R 4.3.2 · C:/Users/petit/Desktop/repos/UO/rok 3/Wprowadzenie do eksploracji danych/lista12/
>
> # Macierz błędów i dokładność dla modelu SVM z funkcją jądra wielomianową
> cat("Model SVM - Wielomianowe jądro:\n")
Model SVM - Wielomianowe jądro:
> print_confusion_matrix_and_accuracy(predictions_polynomial, test_set_standardized$V1)
Macierz błędów:
      predictions
actual -0.100352553066397 0.522962307178777 0.795954419714872 1.2127694128961 1.22907069965322 1.31219636755905
     1                  1                 1                 1               1                1               1
     2                  0                 0                 0               0                0               0
     3                  0                 0                 0               0                0               0
      predictions
actual 1.31544418289391 1.44079553035765 1.49003179725031 1.54786120650768 1.56180336482893 1.5789476361425
     1                0                1                0               1               1              1
     2                1                0                1               0               0              0
     3                0                0                0               0               0              0
      predictions
actual 1.60537252674978 1.62215177270746 1.72115849764271 1.75141222355971 1.84315275758786 1.85431025868786
     1                1                1                0               0               0              0
     2                0                0                1               1               1              1
     3                0                0                0               0               0              0
      predictions
actual 1.8661064205687 1.91990826870208 1.9285076029051 1.95723109710148 1.99134703067364 2.08695898667021
     1               0                0               0               0               0              0
     2               1                1               1               1               1              1
     3               0                0               0               0               0              0
      predictions
actual 2.25736187016896 2.33557353503455 2.38732983683066 2.52277995844221 2.61093514135363 2.66192431523577
     1                0                0                0                0                0              0
     2                1                0                1                0                0              0
     3                0                1                0                1                1              1
      predictions
actual 2.81938869024374 2.84008323495399 3.05892885205332 3.15405979162682 4.6624403142292
     1                0                0                0                0               0
     2                0                0                0                0               0
     3                1                1                1                1               1
Dokładność: 0.02857143
```

```
>
> # Macierz błędów i dokładność dla modelu SVM z funkcją jądra sigmoidalną
> cat("Model SVM - Sigmoidalne jądro:\n")
Model SVM - Sigmoidalne jądro:
> print_confusion_matrix_and_accuracy(predictions_sigmoid, test_set_standardized$V1)
Macierz błędów:
      predictions
actual 0.170958816936003 0.254592249240098 0.562343524885844 0.654949701319578 0.797110155552925 1.02177768725768
     1                 1                 0                 1                 1                 0               1
     2                 0                 1                 0                 0                 1               0
     3                 0                 0                 0                 0                 0               0
      predictions
actual 1.16512169218501 1.17914965115673 1.19346859315737 1.21862561431272 1.22244622412712 1.29340923469568
     1                1                1                1                0                1              0
     2                0                0                0                1                0              1
     3                0                0                0                0                0              0
      predictions
actual 1.39114358031228 1.39472017000781 1.39921570201934 1.44890281801352 1.46071247254648 1.50586735089311
     1                1                0                0                1                1              0
     2                0                1                1                0                0              0
     3                0                0                0                0                0              1
      predictions
actual 1.60140155283522 1.63426976016926 2.0629095414107 2.07475717194818 2.36698007397682 2.38123497925278
     1                1                0                0                0               0              0
     2                0                1                1                0               0              0
     3                0                0                0                1               1              1
      predictions
actual 2.40159199207233 2.47988220563543 2.49648333924844 2.63956539861096 2.65716453524223 2.73353272515553
     1                0                0                0                0                0              0
     2                0                1                1                0                1              0
     3                1                0                0                1                0              1
      predictions
actual 2.77841684375164 2.83009076470705 2.93846495025488 3.03640127260416 3.59408296574455
     1                0                0                0                0               0
     2                0                1                1                1               0
     3                1                0                0                0               1
Dokładność: 0.05714286
>
```

j) Sporządź wykres słupkowy ilustrujący dokładność klasyfikacji dla metod klasyfikacji: Bayes, SVM-f. radialna, SVM-f. liniowa, SVM-f. wielomianowa i SVM-f.sigmoidalna.

```r
# Przykładowe dane dotyczące dokładności (zastąp wartościami uzyskanymi z modeli)
dokladnosci <- data.frame(
  Metoda = c("Bayes", "SVM-Radialna", "SVM-Liniowa", "SVM-Wielomianowa", "SVM-Sigmoidalna"),
  Dokladnosc = c(0.97, 0.95, 0.96, 0.94, 0.93) # przykładowe wartości
)

# Tworzenie wykresu słupkowego
ggplot(dokladnosci, aes(x = Metoda, y = Dokladnosc)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  theme_minimal() +
  labs(title = "Dokładność klasyfikacji różnych metod",
       x = "Metoda",
       y = "Dokładność")
```