PRACE NAUKOWE UNIWERSYTETU EKONOMICZNEGO WE WROCŁAWIU
Nr 7 (1207) 2008
TAKSONOMIA 15
Klasyfikacja i analiza danych – teoria i zastosowania

Marek Walesiak
Uniwersytet Ekonomiczny we Wrocławiu

PROCEDURA ANALIZY SKUPIEŃ Z WYKORZYSTANIEM PROGRAMU KOMPUTEROWEGO CLUSTERSIM I ŚRODOWISKA R

1. Wstęp

W analizie skupień wyodrębnia się siedem etapów (por. np. [Milligan 1996, s. 342-343; Walesiak 2005]): wybór obiektów i zmiennych, wybór formuły normalizacji wartości zmiennych, wybór miary odległości, wybór metody klasyfikacji, ustalenie liczby klas, ocena wyników klasyfikacji, opis (interpretacja) i profilowanie klas.

W artykule scharakteryzowane zostaną funkcje pomocnicze pakietu `clusterSim`[1] oraz wybrane funkcje pakietów `stats`, `cluster` `ade4`, służące poszczególnym etapom analizy skupień. Ponadto zaprezentowane zostaną przykładowe składnie poleceń (procedury) ułatwiające potencjalnemu użytkownikowi realizację wielu zagadnień klasyfikacyjnych niedostępnych w podstawowych pakietach statystycznych (np. SPSS, Statistica, S-Plus, SAS).

2. Etapy w analizie skupień oraz pakiety i funkcje programu R

W tab. 1 pokazano wybrane pakiety i funkcje programu R wykorzystywane w poszczególnych etapach analizy skupień. Szczegółowo zostaną scharakteryzowane funkcje niezbędne z punktu widzenia dalszych rozważań, zawarte w pakietach `clusterSim`, `stats`, `cluster` i `ade4`.

[1] Zob. prace: [Walesiak, Dudek 2006; 2007a; 2007b; Walesiak 2006].

Tabela 1. Etapy w analizie skupień oraz funkcje programu R

Lp.	Etapy w analizie skupień	Wybrane pakiety i funkcje programu R
1	Wybór obiektów i zmiennych	Pakiet `clusterSim` (funkcja `HINoV.Mod`)
2	Wybór formuły normalizacji wartości zmiennych	Pakiet `clusterSim` (funkcja `data.Normalization`)
3	Wybór miary odległości	Pakiet `clusterSim` (funkcje `dist.BC`, `dist.GDM`, `dist.SM`) Pakiet `stats` (funkcja `dist`) Pakiet `ade4` (funkcja `dist.binary`)
4	Wybór metody klasyfikacji	Pakiet `cluster` (funkcja `agnes`, `diana`, `pam`) Pakiet `stats` (funkcje `kmeans`, `hclust`) Pakiet `clusterSim` (funkcja `initial.Centers`)
5	Ustalenie liczby klas	Pakiet `clusterSim` (funkcje `index.G1`, `index.G2`, `index.G3`, `index.S`, `index.KL`, `index.H`, `index.Gap`)
6	Ocena wyników klasyfikacji	Pakiet `clusterSim` (funkcja `replication.Mod`)
7	Opis (interpretacja) i profilowanie klas	Pakiet `clusterSim` (funkcja `cluster.Description`)

Źródło: opracowanie własne.

Etap 1. Wybór obiektów i zmiennych do klasyfikacji

Carmone, Kara i Maxwell [1999] zaproponowali heurystyczną procedurę doboru zmiennych *HINoV* powiązaną z metodą *k*-średnich i skorygowanym indeksem Randa. Celem tej metody jest dobór mniejszej liczby zmiennych z pierwotnego zbioru przez eliminację tych, które zakłócają istniejącą strukturę klas w badanej zbiorowości obiektów. W wyniku zastosowania metody *HINoV* otrzymuje się dla każdej zmiennej jej wkład do istniejącej struktury klas. W pakiecie `clusterSim` znajduje się funkcja prezentująca rozszerzoną wersję tej metody dla danych niemetrycznych i innych metod klasyfikacji (zob. tab. 2).

Tabela 2. Składnia funkcji `HINoV.Mod` w pakiecie `clusterSim`

`HINoV.Mod(x, type="metric", s=2, u, distance=NULL, method="kmeans", Index="cRAND")`	
`x`	macierz danych
`s`	tylko dla danych metrycznych: 1 – ilorazowe; 2 – interwałowe lub mieszane
`u`	liczba klas (dla danych metrycznych)
`distance`	`NULL` dla metody `kmeans` i danych niemetrycznych dla danych ilorazowych: "`d1`" – Manhattan, "`d2`" – Euklidesowa, "`d3`" – Czebyszewa (max), "`d4`" – kwadrat Euklidesowej, "`d5`" – GDM1, "`d6`" – Canberra, "`d7`" – Braya i Curtisa dla danych przedziałowych lub mieszanych: "`d1`", "`d2`", "`d3`", "`d4`", "`d5`"
`method`	`NULL` dla danych niemetrycznych metoda klasyfikacji: "`kmeans`" (default), "`single`", "`complete`", "`average`", "`mcquitty`", "`median`", "`centroid`", "`Ward`", "`pam`"
`Index`	"`cRAND`" – skorygowany indeks Randa, "`RAND`" – indeks Randa

Źródło: opracowanie własne.

Tabela 3. Formuły normalizacyjne dla danych metrycznych

Nr	Nazwa formuły	Formuła	Skala pomiaru zmiennych	
			przed normalizacją	po normalizacji
n0	Bez normalizacji	–	ilorazowa i (lub) przedziałowa	–
n1	Standaryzacja	$z_{ij} = (x_{ij} - \overline{x}_j)/s_j$	ilorazowa i (lub) przedziałowa	przedziałowa
n2	Standaryzacja Webera	$z_{ij} = (x_{ij} - Me_j)/1,4826 \cdot MAD_j$	ilorazowa i (lub) przedziałowa	przedziałowa
n3	Unitaryzacja	$z_{ij} = (x_{ij} - \overline{x}_j)/r_j$	ilorazowa i (lub) przedziałowa	przedziałowa
n4	Unitaryzacja zerowana	$z_{ij} = \left[x_{ij} - \min_i \{x_{ij}\}\right]/r_j$	ilorazowa i (lub) przedziałowa	przedziałowa
n5	Normalizacja w przedziale [–1; 1]	$z_{ij} = (x_{ij} - \overline{x}_j)/\max_i \lvert x_{ij} - \overline{x}_j \rvert$	ilorazowa i (lub) przedziałowa	przedziałowa
n6	Przekształcenia ilorazowe	$z_{ij} = x_{ij}/s_j$	ilorazowa	ilorazowa
n7		$z_{ij} = x_{ij}/r_j$	ilorazowa	ilorazowa
n8		$z_{ij} = x_{ij}/\max_i \{x_{ij}\}$	ilorazowa	ilorazowa
n9		$z_{ij} = x_{ij}/\overline{x}_j$	ilorazowa	ilorazowa
n10		$z_{ij} = x_{ij}/\sum_{i=1}^{n} x_{ij}$	ilorazowa	ilorazowa
n11		$z_{ij} = x_{ij}/\sqrt{\sum_{i=1}^{n} x_{ij}^2}$	ilorazowa	ilorazowa

$x_{ij}(z_{ij})$ – wartość (znormalizowana wartość) j-tej zmiennej dla i-tego obiektu, $\overline{x}_j$ (s_j, r_j) – średnia (odchylenie standardowe, rozstęp) dla j-tej zmiennej, $Me_j(MAD_j)$ – mediana (medianowe odchylenie bezwzględne) dla j-tej zmiennej.

Źródło: opracowanie własne.

Tabela 4. Składnie funkcji pozwalających obliczać miary odległości dla danych metrycznych i niemetrycznych

`dist.GDM (x,method="GDM1")` – funkcja obliczająca macierz odległości według miary GDM (zob. Walesiak [2006]), gdzie `x` – macierz danych, GDM1 – miara odległości GDM dla danych metrycznych, GDM2 – dla danych porządkowych	
`dist.BC (x)` – funkcja obliczająca macierz odległości według formuły Braya-Curtisa dla zmiennych ilorazowych	
`dist.SM(x)` – funkcja obliczająca macierz odległości według miary Sokala-Michenera dla zmiennych nominalnych	
`dist(x, method="euclidean", p = 2)`	
`x`	macierz danych lub obiekt `"dist"`
`method`	miara odległości: `"euclidean"` (odległość euklidesowa), `"maximum"`, `"manhattan"`, `"canberra"`, `"binary"`, `"minkowski"`
`p`	potęga dla odległości `"minkowski"`
`dist.binary(df, method = NULL)`	
`df`	macierz danych zawierająca wartości dodatnie lub zera. Używana z funkcją `as.matrix(1 * (df > 0))`
`method`	liczba od 1 do 10 oznaczająca formułę odległości $d = \sqrt{1-s}$: 1 = Jaccard, 2 = Sokal & Michener, 3 = Sokal & Sneath (1), 4 = Rogers & Tanimoto, 5 = Czekanowski, 6 = Gower & Legendre (1), 7 = Ochiai, 8 = Sokal & Sneath (2), 9 = Phi of Pearson, 10 = Gower & Legendre (2)

Źródło: opracowanie własne.

Tabela 5. Składnie funkcji dla podstawowych metod klasyfikacji w środowisku R*

A. Hierarchiczne metody aglomeracyjne `hclust(d, method = "complete", members=NULL)`	
`d`	macierz odległości
`method`	hierarchiczna metoda aglomeracyjna: `"single"`, `"complete"`, `"ward"`, `"average"`, `"mcquitty"`, `"median"`, `"centroid"`
`members`	NULL or a vector with length size of `d`
B. Metoda *k*-średnich `kmeans(x, centers, iter.max = 10, nstart = 1, algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"))`	
`x`	macierz danych
`centers`	początkowe środki ciężkości klas lub początkowa liczba klas
`iter.max`	maksymalna liczba iteracji
`nstart`	liczba losowych zbiorów danych, jeśli w `centers` podano liczbę klas
`algorithm`	stosowany algorytm
C. Metoda *k*-medoidów `pam(x, k, diss = inherits(x, "dist"), metric = "euclidean", medoids = NULL, trace.lev = 0)`	
`x`	macierz danych lub macierz odległości
`k`	liczba klas
`diss`	dla TRUE x oznacza macierz odległości (domyślnie); dla FALSE – macierz danych
`metric`	miara odległości (x oznacza macierz danych): `"euclidean"`, `"manhattan"`
`medoids`	NULL (domyślnie) lub wektor zawierający początkowe medoidy
`trace.lev`	zawartość drukowanych informacji w różnych fazach algorytmu: 0 (domyślnie) – bez drukowania; wyższa liczba oznacza więcej drukowanych informacji
D. Hierarchiczne metody aglomeracyjne `agnes(x, diss = inherits(x, "dist"), metric = "euclidean", method = "average", par.method)`	
`method`	metoda klasyfikacji: `"average"`, `"single"`, `"complete"`, `"ward"`, `"weighted"`, `"flexible"`
`par.method`	wektor zawierający parametry dla metody `"flexible"` (1, 3 lub 4)
E. Hierarchiczna metoda deglomeracyjna `diana(x, diss = inherits(x, "dist"), metric = "euclidean")`	

* opis niektórych funkcji zawiera podstawowe argumenty.

Źródło: opracowanie własne.

Etap 2. Wybór formuły normalizacji wartości zmiennych

Normalizację wartości zmiennych przeprowadza się w pakiecie `clusterSim` z wykorzystaniem funkcji `data.Normalization (x, type="n0")`, gdzie x oznacza macierz danych, a `type` – typ formuły normalizacyjnej z tab. 3.

Etap 3. Wybór miary odległości

Podstawowe miary odległości, uzależnione od skali pomiaru zmiennych, zawarte są w pakietach `clusteSim`, `stats` i `ade4` (zob. tab. 4).

Etap 4. Wybór metody klasyfikacji

Najczęściej wykorzystywane w badaniach empirycznych metody klasyfikacji znajdują się w pakietach `stats` i `cluster` (zob. tab. 5).

Tabela 6. Indeksy oceny jakości klasyfikacji służące wyborowi liczby klas

Nazwa indeksu / Formuła	Składnia funkcji	Kryterium wyboru liczby klas $\hat{u}$
Calińskiego i Harabasza $G1(u) = \frac{B_u/(u-1)}{W_u/(n-u)}$	`index.G1 (x,cl)` x – macierz danych cl – wektor liczb całkowitych informujących o przynależności obiektów do klas	$\arg\max_u \{G1(u)\}$
Bakera i Huberta $G2(u) = \frac{s(+)-s(-)}{s(+)+s(-)}$	`index.G2(d,cl)` d – macierz odległości	$\arg\max_u \{G2(u)\}$
Huberta i Levine'a $G3(u) = \frac{D(u)-r\cdot D_{min}}{r\cdot D_{max}-r\cdot D_{min}}$	`index.G3(d,cl)`	$\arg\min_u \{G3(u)\}$
Silhouette $S(u) = \frac{1}{n}\sum_{i=1}^{n}\frac{b(i)-a(i)}{\max\{a(i);b(i)\}}$	`index.S(d,cl)`	$\arg\max_u \{S(u)\}$
Krzanowskiego i Lai $KL(u) = \left\lvert\frac{DIFF_u}{DIFF_{u+1}}\right\rvert$ $DIFF_u = (u-1)^{2/m}W_{u-1} - u^{2/m}W_u$	`index.KL (x,clall)` clall – trzy wektory liczb całkowitych informujących o przynależności obiektów do klas w podziale na $u-1$, u i $u+1$ klas	$\arg\max_u \{KL(u)\}$
Hartigana $H(u) = \left(\frac{W_u}{W_{u+1}}-1\right)(n-u-1)$	`index.H (x,clall)` clall – dwa wektory liczb całkowitych informujących o przynależności obiektów do klas w podziale na u i $u+1$ klas	najmniejsze u, dla którego $H(u) \le 10$
Gap $Gap(u) = \frac{1}{B}\sum_{b=1}^{B}\log W_{ub} - \log W_u$ $diffu = Gap(u) - Gap(u+1) + s_{u+1}$ $s_u = sd_u\sqrt{1+1/B}$ sd_u – odchylenie standardowe z wartości $\{\log W_{ub}\}$	`index.Gap (x, clall, reference.distribution="unif", B=10, method="pam")` clall – dwa wektory liczb całkowitych informujących o przynależności obiektów do klas w podziale na u i $u+1$ klas reference.distribution – sposób generowania obserwacji na zmiennych z rozkładu jednostajnego (unif, pc) B=10 – liczba generowanych zbiorów obserwacji method – metoda klasyfikacji (ward, single, complete, average, mcquitty, median, centroid, pam, k-means)	najmniejsze u, dla którego $diffu \ge 0$

$\mathbf{B}_u$ – macierz kowariancji międzyklasowej, $\mathbf{W}_u$ – macierz kowariancji wewnątrzklasowej, tr – ślad macierzy, $B_u(W_u) = tr\mathbf{B}_u\left(tr\mathbf{W}_u\right)$, $r,s = 1,\ldots,u$ – numer klasy, u – liczba klas, $i,k = 1,\ldots,n$ – numer obiektu, n – liczba obiektów, m – liczba zmiennych, $s(+)$ – liczba par odległości zgodnych, $s(-)$ – liczba par odległości niezgodnych, $D(u)$ – suma wszystkich odległości wewnątrzklasowych, r – liczba odległości wewnątrzklasowych, D_{min} (D_{max}) – najmniejsza (największa) odległość wewnątrzklasowa, $a(i) = \sum_{k\in\{P_r\setminus i\}} d_{ik} \Big/ (n_r - 1)$ – średnia odległość obiektu i od pozostałych obiektów należących do klasy P_r; $b(i) = \min_{s\neq r}\{d_{iP_s}\}$, $d_{iP_s} = \sum_{k\in P_s} d_{ik} \Big/ n_s$ – średnia odległość obiektu i od obiektów należących do klasy P_s, B – liczba generowanych zbiorów obserwacji.

Źródło: opracowanie własne na podstawie prac: [Caliński, Harabasz 1974; Hubert 1974; Milligan, Cooper 1985; Kaufman, Rousseeuw 1990; Hartigan 1975; Tibshirani, Walther, Hastie 2001].

Dla metody *k*-średnich początkowe środki ciężkości klas można ustalić za pomocą funkcji `initial.Centers (x, k)` w pakiecie `clusterSim` (`x` – macierz danych, `k` - liczba klas).

Etap 5. Ustalenie liczby klas

Charakterystykę siedmiu indeksów jakości klasyfikacji dostępnych w pakiecie `clusterSim`, a służących wyborowi liczby klas, zawiera tab. 6.

Etap 6. Ocena wyników klasyfikacji

Procedura służąca ocenie wyników klasyfikacji (analiza replikacji – zob. prace: [Breckenridge 2000; Walesiak 2007]) zawarta jest w pakietach `clusterSim` (zob. tab. 7). Replikacja dotyczy przeprowadzenia procesu klasyfikacji zbioru obiektów na podstawie dwóch prób wylosowanych z danego zbioru danych, a następnie ocenie zgodności otrzymanych rezultatów. Poziom zgodności wyników dwóch podziałów (skorygowany indeks Randa) odzwierciedla poziom stabilności przeprowadzonej klasyfikacji zbioru obiektów.

Tabela 7. Analiza replikacji z wykorzystaniem funkcji `replication.Mod` pakietu `clusterSim`

`replication.Mod(x, v="m", u=2, centrotypes="centroids", normalization=NULL, distance=NULL, method="kmeans", S=10, fixedAsample=NULL)`	
`x`	macierz danych
`v`	typ danych: metryczne ("r" – ilorazowe, "i" – przedziałowe, "m" – mieszane), niemetryczne ("o" – porządkowe, "n" – nominalne wielostanowe, "b" – binarne)
`u`	liczba klas
`centrotypes`	"centroids", "medoids"
`normalization`	formuły normalizacyjna dla danych metrycznych `n0-n11`
`distance`	NULL dla metody *k*-średnich ("kmeans"), dla danych ilorazowych: "d1" - Manhattan, "d2" – Euclidean, "d3" – Chebychev (max), "d4" – squared Euclidean, "d5" – GDM1, "d6" – Canberra, "d7" – Bray-Curtis dla danych przedziałowych i mieszanych: "d1", "d2", "d3", "d4", "d5" dla danych porządkowych: "d8" – GDM2 dla danych nominalnych wielostanowych: "d9" – Sokal & Michener dla danych binarnych: "b1" = Jaccard; "b2" = Sokal & Michener; "b3" = Sokal & Sneath (1); "b4" = Rogers & Tanimoto; "b5" = Czekanowski; "b6" = Gower & Legendre (1); "b7" = Ochiai; "b8" = Sokal & Sneath (2); "b9" = Phi of Pearson; "b10" = Gower & Legendre (2)
`method`	metoda klasyfikacji (zob. etap 4)
`S`	liczba symulacji
`fixedAsample`	numery obiektów dobrane losowo do podzbioru `A` (`NULL`) lub numery obiektów dobrane arbitralnie do podzbioru `A`

Źródło: opracowanie własne.

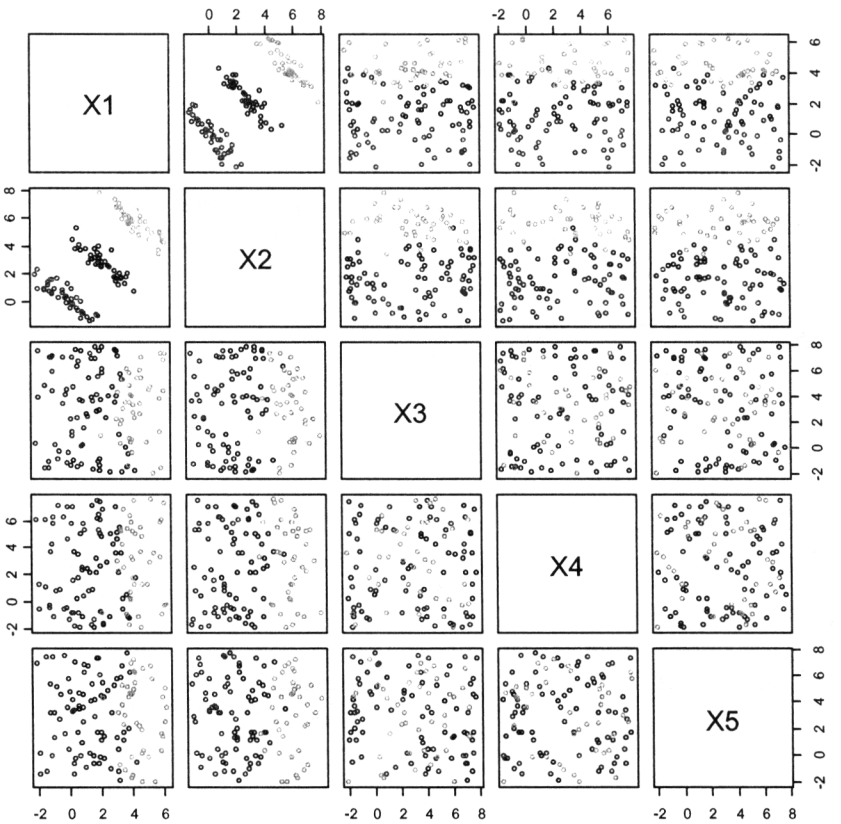
Etap 7. Opis (interpretacja) i profilowanie klas

Do wyznaczenia charakterystyk poszczególnych klas można wykorzystać z pakietu `clusterSim` funkcję `cluster.Description(x, cl, sdType="sample")`. Funkcja ta oblicza osobno dla każdej klasy i zmiennej z ustalo-

nego podziału zbioru obiektów na klasy `cl` następujące statystyki opisowe: średnia arytmetyczna (1), odchylenie standardowe (2), mediana (3), medianowe odchylenie bezwzględne (4), dominanta (5) (dla zmiennych nominalnych i porządkowych, jeśli występuje więcej wartości o maksymalnej częstości występowania zwracana jest wartość „N.A."). W odchyleniu standardowym w mianowniku występuje $n - 1$ dla próby (`sdType="sample"`) i n dla populacji (`sdType="population"`).

3. Przykładowe składnie poleceń z wykorzystaniem wybranych funkcji programu R

Za pomocą dwuwymiarowej zmiennej losowej o rozkładzie normalnym wygenerowano po 40 obserwacji dla trzech skupień o wydłużonym kształcie. Przyjęto



Rys. 1. 120 obserwacji pięciu zmiennych w układach dwuwymiarowych
Źródło: opracowanie własne.

następujące wektory wartości oczekiwanych dla skupień (0, 0), (2, 3), (4; 6) oraz identyczne macierze kowariancji Σ ($\sigma_{jj} = 1$, $\sigma_{jl} = -0{,}9$). Do analizy wprowadzono dodatkowe trzy zmienne zakłócające istniejącą w układzie dwuwymiarowym strukturę klas (tzw. *noisy variables*). Po 120 obserwacji na tych zmiennych wygenerowano niezależnie z rozkładu jednostajnego (zob. rys. 1).

Do wygenerowania danych wykorzystano funkcję `cluster.Gen` pakietu `clusterSim` z następującą składnią poleceń (zob. przykład 1).

Przykład 1

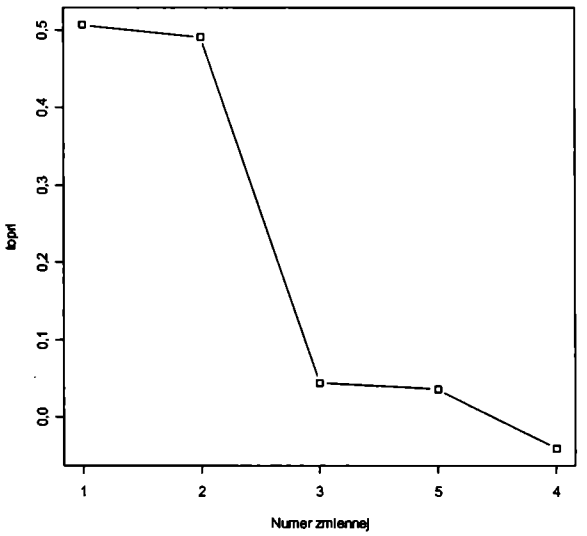
```
> library(clusterSim)
> means <- matrix(c(0,2,4,0,3,6),3,2)
> cov <- matrix(c(1,-0.9,-0.9,1),2,2)
> grnd <- cluster.Gen(numObjects=40, means=means,
                      cov=cov, model=2, numNoisyVar=3)
> colornames <- c("red","blue","green")
> grnd$clusters[grnd$clusters==0]<-length(colornames)
> plot(grnd$data, col=colornames[grnd$clusters])
> write.table(grnd$data, file="F:/dane.csv", sep=";",
              dec=",")
```

W składni poleceń w przykładzie 2 do usunięcia zmiennych zakłócających strukturę klas zastosowano metodę *HINoV* z następującymi parametrami (liczba klas: 3, miara odległości: kwadrat odległości euklidesowej d4, metoda klasyfikacji: pam).

Przykład 2

```
> library(cluster)
> library(clusterSim)
> x <- read.csv2("dane.csv", header=TRUE, row.names=1)
> x <- as.matrix(x)
> x <- as.data.frame(x)
> r1 <- HINoV.Mod(x, type="metric", s=2, 3, dis-
tance="d4", method="pam",Index="cRAND")
> options(OutDec = ",")
> plot(r1$stopri[,2],type="b", pch=0, xlab="Numer
zmiennej", ylab="topri",xaxt="n")
>
axis(1,at=c(1:max(r1$stopri[,1])),labels=r1$stopri[,1])
```

W wyniku zastosowania tej procedury otrzymuje się wykres osypiska (rys. 2).



Rys. 2. Wykres osypiska

Źródło: opracowanie własne.

Na podstawie wykresu osypiska zmienne o numerach 3, 4 i 5, zakłócające istniejącą w układzie dwuwymiarowym strukturę klas, zostają usunięte za pomocą metody *HINoV*.

W składni poleceń w przykładzie 3 przyjęto następujące założenia:

- obserwacje na zmiennych poddano standaryzacji Webera (n2),
- do pomiaru odległości między obiektami zastosowano kwadrat odległości euklidesowej,
- do podziału zbioru 120 obiektów opisanych zmiennymi V1 i V2 na klasy zastosowano metodę *k*-medoidów (pam),
- liczbę klas ustalono na podstawie indeksu gap (index.Gap),
- za pomocą instrukcji write.table zapisujemy w plikach wartości „diffu" służące wyborowi liczby klas dla indeksu gap, zaklasyfikowanie obiektów do klas („clustering") oraz wartości podstawowych parametrów informujących o otrzymanych klasach obiektów („clusinfo").

Przykład 3

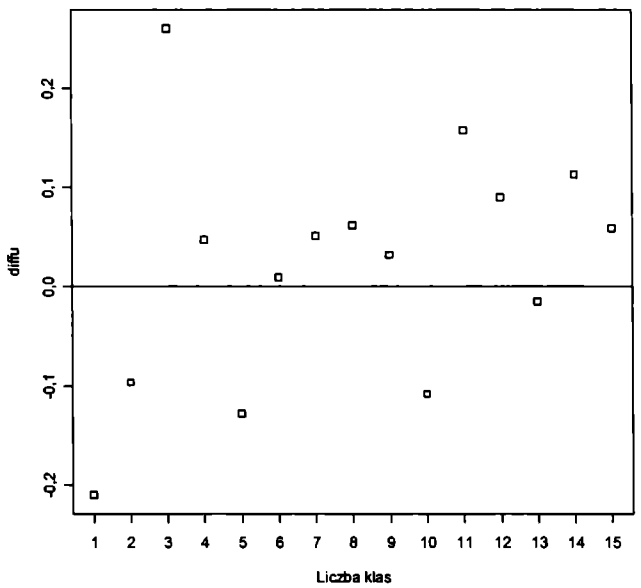
```
> library(clusterSim)
> library(cluster)
> x <- read.csv2("c:/Dane_120x2.csv", header=TRUE,
row.names=1)
> x <- as.matrix(x)
> z <- data.Normalization (x, type="n2")
```

```
> z <- as.data.frame(z)
> d <- dist(z, method="euclidean")^2
> min_liczba_klas=1
> max_liczba_klas=15
> min<- 0
> wyniki<- array(0,c(max_liczba_klas-min_liczba_klas+1,
2))
> wyniki[,1]<- min_liczba_klas:max_liczba_klas
> znaleziono<-FALSE
> for (liczba_klas in min_liczba_klas:max_liczba_klas)
> {
> cl1 <- pam(d, liczba_klas, diss = TRUE)
> cl2 <- pam(d, liczba_klas+1, diss = TRUE)
> clall<- cbind(cl1$clustering,cl2$clustering)
> sink(file="tymczasowy.txt")
> Gap <- index.Gap(z, clall, reference.distribution =
"pc", B=10, method="pam")
> wyniki[liczba_klas - min_liczba_klas+1,2]<-diffu<-
print(Gap$diffu)
> sink()
> if ((wyniki[liczba_klas - min_liczba_klas+1,2]>=0) &&
(!znaleziono))
> {
> lk<- liczba_klas
> min<-diffu
> clopt<-cl1$cluster
> wyn<-cl1$clusinfo
> znaleziono<-TRUE
> }
> }
> if (znaleziono)
> {
> print(paste("minimalna liczba klas dla diffu>=0 wyno-
si", lk, "dla diffu=",min))
> }else
> {
> print("Nie znalazłem klasyfikacji, dla której
diffu>=0")
> }
> write.table(wyniki, file="diffu.csv", sep=";",
dec=",", row.names=TRUE, col.names=FALSE)
> write.table(clopt, file="clustering.csv", sep=";",
dec=",", row.names=TRUE, col.names=FALSE)
```

```
> write.table(wyn, file="clusinfo.csv", sep=";",
dec=",", row.names=TRUE, col.names=TRUE)
> options(OutDec = ",")
> plot(wyniki, type="p", pch=0, xlab="Liczba klas",
ylab="diffu",xaxt="n")
> abline(h=0, untf = FALSE)
> axis(1,c(min_liczba_klas:max_liczba_klas))
```

W wyniku zastosowania tej procedury otrzymuje się podział zbioru 120 obiektów opisanych dwiema zmiennymi X1 i X2 na trzy klasy:

```
[1] "minimalna liczba klas dla diffu>=0 wynosi 3 dla
diffu= 0,260407103676564"
```



Rys. 3. Graficzna prezentacja wartości indeksu gap
Źródło: opracowanie własne.

W przykładzie 4 przeprowadzono ocenę otrzymanych wyników klasyfikacji, wykorzystując funkcję `replication.Mod` z pakietu `clusterSim`.

Przykład 4

```
> library(clusterSim)
> x <- read.csv2("Dane_120x2.csv", header=TRUE,
row.names=1)
> x <- as.matrix(x)
```

```
> options(OutDec = ",")
> w<- replication.Mod(x, v="m", u=3, centro-
types="centers", normalization="n2", distance="d4",
method="pam", S=50)
> print(w$cRand)
```

W wyniku zastosowania tej procedury otrzymuje się:

```
[1] 0,9297423
```

Zatem wartość skorygowanego indeksu Randa świadczy o wysokiej stabilności podziału zbioru 120 obiektów na trzy klasy.

4. Podsumowanie

W artykule zaprezentowano siedem etapów typowej procedury analizy skupień, a następnie scharakteryzowano funkcje pakietów `clusterSim`, `stats`, `cluster` i `ade4` środowiska R służących realizacji poszczególnych jej etapów. Ponadto przedstawiono przykładowe składnie poleceń (procedury) z wykorzystaniem analizowanych funkcji, ułatwiające potencjalnemu użytkownikowi realizację wielu zagadnień klasyfikacyjnych niedostępnych w podstawowych pakietach statystycznych.

Literatura

Breckenridge J.N. (2000), *Validating Cluster Analysis: Consistent Replication and Symmetry*, „Multivariate Behavioral Research", 35 (2), s. 261-285.

Caliński R.B., Harabasz J. (1974), *A Dendrite Method for Cluster Analysis*, „Communications in Statistics", vol. 3, s. 1-27.

Carmone F.J., Kara A., Maxwell S. (1999), *HINoV: a New Method to Improve Market Segment Definition by Identifying Noisy Variables*, „Journal of Marketing Research", November, vol. 36, s. 501-509.

Hartigan J. (1975), *Clustering Algorithms*, Wiley, New York.

Hubert L. (1974), *Approximate Evaluation Technique for the Single-link and Complete-link Hierarchical Clustering Procedures*, „Journal of the American Statistical Association", vol. 69, nr 347, s. 698-704.

Kaufman L., Rousseeuw P.J. (1990), *Finding Groups in Data: an Introduction to Cluster Analysis*, Wiley, New York.

Milligan G.W. (1996), *Clustering Validation: Results and Implications for Applied Analyses*, [w:] *Clustering and Classification*, red. P. Arabie, L.J. Hubert, G. de Soete, World Scientific, Singapore, s. 341-375.

Milligan G.W., Cooper M.C. (1985), *An Examination of Procedures of Determining the Number of Cluster in a Data Set*, „Psychometrika", vol. 50, nr 2, s. 159-179.

R Development Core Team (2006), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, URL http://www.R-project.org.

Tibshirani R., Walther G., Hastie T. (2001), *Estimating the Number of Clusters in a Data Set via the Gap Statistic*, „Journal of the Royal Statistical Society", ser. B, vol. 63, part 2, s. 411-423.

Walesiak M. (2005), *Rekomendacje w zakresie strategii postępowania w procesie klasyfikacji zbioru obiektów*, [w:] *Przestrzenno-czasowe modelowanie i prognozowanie zjawisk gospodarczych*, red. A. Zeliaś, AE, Kraków, s. 185-203.

Walesiak M. (2006), *Uogólniona miara odległości GDM w programie komputerowym clusterSim dla środowiska R*, [w:] Taksonomia 14, red. K. Jajuga, M. Walesiak, Prace Naukowe Akademii Ekonomicznej we Wrocławiu nr 1169, AE, Wrocław.

Walesiak M., Dudek A. (2006), *Symulacyjna optymalizacja wyboru procedury klasyfikacyjnej dla danego typu danych – oprogramowanie komputerowe i wyniki badań*, [w:] Taksonomia 13, red. K. Jajuga, M. Walesiak, Prace Naukowe Akademii Ekonomicznej we Wrocławiu nr 1126, AE, Wrocław, s. 120-129.

Walesiak M., Dudek A. (2007a), *Symulacyjna optymalizacja wyboru procedury klasyfikacyjnej dla danego typu danych – charakterystyka problemu*, Zeszyty Naukowe Uniwersytetu Szczecińskiego nr 450, s. 635-646.

Walesiak M., Dudek A. (2007b), *Determination of Optimal Clustering Procedure for a Data Set*, 30th Annual Conference of the German Classification Society (GfKl) „Advances in Data Analysis", Berlin, March 8-10, 2006.

Walesiak M. (2007), *Ocena stabilności wyników klasyfikacji z wykorzystaniem analizy replikacji*, Prace Naukowe Akademii Ekonomicznej we Wrocławiu (w przygotowaniu).

CLUSTER ANALYSIS PROCEDURE WITH CLUSTERSIM COMPUTER PROGRAMME AND R ENVIRONMENT

Summary

The first part of the article presents major steps in a cluster analysis procedure (see [Milligan 1996, 342-343; Walesiak 2005]). The next part presents the functions of `clusterSim`, `stats`, `cluster`, and `ade4` packages of R environment which are applied to solving clustering problems in each stage of this procedure. Also the examples of the syntax (procedures) for solving different clustering problems are presented. These procedures help to resolve a broad range of classification problems that are not available in statistical packages (e.g. SPSS, Statistica, S-Plus, SAS).