Funkcja dokonująca porządkowania za pomocą metody Hellwiga

Wstęp

Na początku użytkownik przed skorzystaniem z funkcji musi zaimportować dane, które chce poddać porządkowaniu. W tym celu może użyć np. takiego wywołania z podaniem swojej ścieżki pliku formatu xlsx, który chce poddać porządkowaniu. Jako przykład zaimportowałam tabele zawierającą 8 obiektów, będących ofertami sprzedaży aut.

```
library(readxl)
zbior_danych <- read_excel("datasets/8_Rozniacych_sie_obiektow.xlsx",
                           sheet = "Arkusz1", col_types = c("numeric","text",
                           "text", "text", "text", "text",
                              "numeric", "numeric", "text",  "numeric",
                              "text", "text", "text", "text","text",
                                  "text", "numeric", "text", "text",
                                   "text", "numeric", "numeric",
                                   "numeric", "numeric", "numeric"))
```

```
## `col_type = "blank"` deprecated. Use "skip" instead.
```

Podgląd danych:

```
head(zbior_danych)
```

```
## # A tibble: 6 x 29
##      Nr MARKA  MODEL  WERSJA TYP   WOJEWODZTWO `CENA.BRUTTO_[p~ `MOC_[km]`
##   <dbl> <chr>  <chr>  <chr>  <chr> <chr>                  <dbl>      <dbl>
## 1  1.00 Mazda  3      II     komp~ zachodniop~            25200        105
## 2  2.00 Jaguar XF     X260   kombi dolnoslask~           323600        240
## 3  3.00 Subaru B9 Tr~ <NA>   suv   malopolskie            38900        245
## 4  4.00 Volks~ Golf   VII    kombi lodzkie               113900        150
## 5  5.00 Peuge~ 508    <NA>   kombi slaskie                42500        115
## 6  6.00 Opel   Antara <NA>   suv   lodzkie                24000        150
## # ... with 21 more variables: `POJEMNOSC.SKOKOWA_[cm3]` <dbl>,
## #   ROK.PRODUKCJI <dbl>, `PRZEBIEG_[km]` <dbl>, KOLOR <chr>, L.DZRZWI
## #   <dbl>, RODZAJ.PALIWA <chr>, SKRZYNIA.BIEGOW <chr>, NAPED <chr>,
## #   KRAJ.AKTUALNEJ.REJESTRACJI <chr>, KRAJ.POCHODZENIA <chr>,
## #   STATUS.POJAZDU.SPROWADZONEGO <chr>, PIERWSZY.WLASCICIEL <dbl>,
## #   KTO.SPRZEDAJE <chr>, STAN <chr>, SERWISOWANY <chr>, ABS <dbl>,
## #   KOMPUTER.POKLADOWY <dbl>, ESP <dbl>, KLIMATYZAJCA <dbl>, BEZWYPADKOWY
## #   <dbl>, USZKODZONY <dbl>
```

Podzbiór danych

W kolejnym, kroku po przyjrzeniu się zbiorowi danych, użytkownik musi zadecydować na których danych ilościowych chce pracować - ważna jest znajomość danych. Dodatkowo pierwszą kolumną musi być kolumna zawierająca numery indeksów obiektów, ze względu na to, że w wyniku zastosowania funkcji odpowiedzialnej za porządkowanie, zostaną zwrócone w kolejności malejącej numery indeksów, mówiące o kolejności uporządkowania. W związku z tym, za pomocą poniższej procedury użytkownik tworzy podzbiór zaimportowanego

zbioru, gdzie w miejsce "" wpisuje nazwy kolumn zawierających zmienne ilościowe, wybrane do porządkowania(przyjmijmy założenie, że podzbiór będzie nazywał się dane_porzadkowanie - będzie to pomocne w dalszej części programu). U mnie wybranymi kolumnami są: cena, moc, pojemność, rok produkcji, przebieg.

```
dane_porzadkowanie<-zbior_danych[c("Nr","CENA.BRUTTO_[pln]","MOC_[km]",
                                   "POJEMNOSC.SKOKOWA_[cm3]",
                                   "ROK.PRODUKCJI","PRZEBIEG_[km]")]
```

Transformacje danych

Chcąc zastosować metodę Hellwiga do uporządkowania zbioru, wymaganym jest aby zmienne miały charakter stymulant oraz by zostały poddane transformacji normalizacyjnej. Aby funckja dokonująca porządkowania dawała poprawny wynik, użutkownik musi zająć się tranformacją przed jej zastosowaniem. Poniżej podałam tego przykład. Dla zmiennych które stymulantami nie są, należy dokonać stymulacji. Wsród moich zmiennych poddanych porządkowaniu, do stymulant nie należy zmienna zmienna: przebieg - jest destymulantą, w związku z tym, została przekształcona na stymulante, za pomocą przekształcenia ilorazowego podanego w postaci funkcji:

```
stymulacja_przeksztalcenie_ilorazowe<-function(x,y){
  for (i in 1:nrow(x)){
    x[i,which(colnames(x)==y)]=1/x[i,which(colnames(x)==y)]
  }
  return(x)
}
```

Gdy użytkownik chce skorzystać z tej funkcji, w miejsce x musi wpisać nazwę zbioru, a w miejsce y nazwę kolumny w "", którą chce poddać stymulacji. UWAGA - kolumny wymagające stymulacji, muszą zostać osobno poddane działaniu poniższej funkcji, dodatkowo po każdym zastosowaniu funkcji, należy nadpisać zbiór by zmiany zostały zapisane.

```
dane_porzadkowanie<-stymulacja_przeksztalcenie_ilorazowe(dane_porzadkowanie, "PRZEBIEG_[km]")
```

W celu uzyskania porównywalności między zmiennymi, zostały one poddane transofrmacji normalizacyjnej - standaryzacji, dzięki temu każda zmienna uzyskuje średnią wartość oczekiwaną równą 0, a odchylenie standardowe równe 1. Poniżej została zaprezentowana funkcja dokonująca standaryzacji.

```
standaryzacja<-function(x){
      suma=0
      srednia=0
      odchylenie=0
      for (j in 2:ncol(x)){
        suma[j]=sum(x[j])
        srednia[j]=suma[j]/nrow(x)
        suma_kwadratow=0
        kwadrat=0
        for(i in 1:nrow(x)){
          kwadrat=(x[i,j]-srednia[j])^2
          suma_kwadratow=suma_kwadratow+kwadrat
        }
        odchylenie[j]=sqrt(suma_kwadratow/nrow(x))
        for (i in 1:nrow(x)){
          x[i,j]=(x[i,j]-srednia[j])/odchylenie[j]
        }
      }
      return(x)
```

```
    }
```

Funkcja porządkująca metodą Hellwiga

W celu dokonania porządkowania na unormowanych danych, należy zastosować poniższą funkcję tj. metoda_Hellwiga, która jest postaci:

```
metoda_Hellwiga<-function(x){
  x<-standaryzacja(x) #standaryzacja zbioru stymulant
#wyznaczenie obiektu wzorcowego, zmienne maja charakter stymulant wiec wspolrzedne
#obiekt_wz = max_j
obiekt_wz=0
for (j in 2:ncol(x)){  #od 2kolumny bo 1 kolumna to Nr - index
  obiekt_wz[j]=max(x[j])
    }

## odleglosci od kazdego obiektu
odleg<- x[c("Nr" )]
for (i in 1:nrow(x)){
  SUMKA=0
  for (j in 2:ncol(x)){
    SUMKA=SUMKA+(x[i,j]-obiekt_wz[j])^2
  }
  odleg[i,2]=sqrt(SUMKA) #kolumna zawierajaca odleglosci
}

#odchylenie standardowe dla odleglosci - moze tez tu mozna skorzystac z funkcji z tym
#ze dla innego argumentu
d_0=0
suma=0
srednia=0
odchylenie=0
for (j in 2:ncol(odleg)){
  suma[j]=sum(odleg[j])
  srednia[j]=suma[j]/nrow(odleg)
  suma_kwadratow=0
  kwadrat=0
  for(i in 1:nrow(odleg)){
    kwadrat=(odleg[i,j]-srednia[j])^2
    suma_kwadratow=suma_kwadratow+kwadrat
  }

  odchylenie[j]=sqrt(suma_kwadratow/nrow(odleg))
  d_0=srednia[j]+2*odchylenie[j]  #d_0 to po prostu wartosc
}

#ostatnia kolumna to jak zawsze zmienna syntetyczna
    x[,"zmienna_syntetyczna"] <-0
        for (i in 1:nrow(x)){
      x[i,ncol(x)]=1-(odleg[i,2]/d_0)
        }
x<-x[order(-x$zmienna_syntetyczna),]
return(x[1])
```

```
}
```

Wywołanie funkcji dla zbioru dane_porzadkowanie - podzbioru wyjściowych danych

```
metoda_Hellwiga(dane_porzadkowanie)
```

```
## # A tibble: 8 x 1
##      Nr
##   <dbl>
## 1  2.00
## 2  4.00
## 3  3.00
## 4  6.00
## 5  5.00
## 6  1.00
## 7  8.00
## 8  7.00
```

Na podstawie powyższego wyniku, widać, że 1-wsze miejsce zajęła oferta sprzedaży z numerem indeksu = 2, a ostatnie oferta z numerem indeksu o numerzer 7.