Funkcja dokonująca porządkowania za pomocą metody rang

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("~/Praca licencjacka/Moje dane_+zrodlo/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 rang 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:

```
for (i in 1:nrow(dane_porzadkowanie)){
  dane_porzadkowanie[i,which(colnames(dane_porzadkowanie)=="PRZEBIEG_[km]")]=
    1/dane_porzadkowanie[i,which(colnames(dane_porzadkowanie)=="PRZEBIEG_[km]")]
}
```

W celu uzyskania porównywalności między zmiennymi, zostały one poddane transofrmacji normalizacyjnej - unitaryzacji, dzięki temu wartości każdej zmiennej należą do przedziału [0,1]

```
maksi=0
minim=0
for (j in 2:ncol(dane_porzadkowanie)){
  maksi[j]=max(dane_porzadkowanie[j])
  minim[j]=min(dane_porzadkowanie[j])
  for (i in 1:nrow(dane_porzadkowanie)){
    dane_porzadkowanie[i,j]=(dane_porzadkowanie[i,j]-minim[j])/(maksi[j]-minim[j])
  }
}
```

Funkcja porządkująca metodą rang

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

```
funkcja_porzadkowanie_metoda_rang<-function(x){
  y<-x #dzieki temu nie bede sztywno odwolywac sie do 1kolumny rang
  for (i in 2:ncol(x)){
    x[ncol(x)+1]=rank(-x[i])
  }
  #ostania kolumna to zmienna_syntetyczna - za pomoca metody sredniej arytmetycznej
    x[,"zmienna_syntetyczna"] <-0
      for(i in 1:nrow(x)){
        for(j in (ncol(y)+1):(ncol(x)-1)){
          x[i,ncol(x)]=x[i,ncol(x)]+x[i,j]
            j=j+1
      }
    x[i,ncol(x)]=x[i,ncol(x)]/(ncol(x)-7)
    }
```

```
  x<-x[order(x$zmienna_syntetyczna),]
  print("Numery indeksów obiektów po uporządkowaniu: ")
  return(x[1])
}
```

Wywołanie funkcji dla zbioru dane_porzadkowanie - podzbioru wyjściowych danych

```
funkcja_porzadkowanie_metoda_rang(dane_porzadkowanie)
```

```
## [1] "Numery indeksów obiektów po uporządkowaniu: "
```

```
## # A tibble: 8 x 1
##      Nr
##   <dbl>
## 1  2.00
## 2  4.00
## 3  3.00
## 4  5.00
## 5  6.00
## 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.