Dzień 1 - Wprowadzenie do R - zadania

Patryk Czortek, Marcin K. Dyderski

1 kwietnia 2019

Zadania do wykonania

1. Załóżmy, że dysponujemy próbkami masy ciała, koloru sierści oraz struktury płci z populacji ryjówki aksamitnej. Dane reprezentujące próbę można wpisać w skrypt w następujący sposób:

```
r shrews.mass<-c(26,29,41,24,28,56,74,35,68,95,45,
67,89,35,67,88,75,34)

fur.color<-c('gray','gray','gray','gray','brown','brown',
'brown','brown','brown','black','black','black',
'black','black','black','gray','brown','brown')

male<-c('TRUE','TRUE','TRUE','TRUE','FALSE','FALSE',
        'FALSE','FALSE','FALSE','TRUE','TRUE','TRUE','TRUE',
        'FALSE','FALSE','FALSE','FALSE','FALSE')
```

   a) Korzystając z funkcji `mean()` obliczyć średnią arytmetyczną z masy ciała ryjówek Przyjmując założenie, że dane masy ciała reprezentują rozkład normalny:
   b) Korzystając z funkcji `var()` obliczyć wariancję
   c) Korzystając z funkcji `sd()` obliczyć odchylenie standardowe
   d) Dane masy ciała ryjówek zapisać jako ciąg liczb (funkcja `as.integer()`), kolor sierści jako ciąg znaków (funkcja `as.character()`), a strukturę płci jako wartość logiczną (funkcja `as.logical()`)
   e) Z trzech wektorów stworzonych w podpunkcie (d) stworzyć listę (funkcja `list()`). Każdemu poziomowi listy nadać nazwę (funkcja `names()`)
   f) Listę przekształcić w ramkę danych (funkcja `as.data.frame()`)

2. W plikach `molinion1.csv` oraz `molinion2.csv` zawarto dane procentowego pokrycia gatunków roślin naczyniowych na n powierzchniach badawczych zlokalizowanych na łąkach zmiennowilgotnych w okolicach Dąbrowy Górniczej. Linki: [https://github.com/mkdyderski/BSS/blob/BSS2019/datasety/molinion1.csv] oraz [https://github.com/mkdyderski/BSS/blob/BSS2019/datasety/molinion2.csv]. Możesz również ściągnąć go do R za pomocą funkcji `read.csv()`:

```
moli1<-read.csv('https://raw.githubusercontent.com/mkdyderski/BSS/BSS2019/datasety/molinion1.csv',
                sep=';')
moli2<-read.csv('https://raw.githubusercontent.com/mkdyderski/BSS/BSS2019/datasety/molinion2.csv',
                sep=';')
```

   a) Oba pliki wczytać do R
   b) Sprawdzić, czy dane zostały wczytane poprawnie (funkcja `head()`, `str()`, `dim()` oraz `length()`)
   c) Dokonać transpozycji obu ramek danych (funkcja `t()`)
   d) Korzystając z biblioteki `reshape2` obie ramki danych przekształcić do wąskich tabelek (funkcja `melt()`)
   e) Obie wąskie tabelki skleić wierszami w jeden obiekt (funkcja `rbind()`)
   f) Zmienić nazwy kolumn. Kolumna pierwsza reprezentuje id powierzchni, druga gatunek, a trzecia procentowe pokrycie gatunku
   g) Korzystając z biblioteki `reshape2` tabelkę wąską przekształcić w szeroką (funkcja `d.cast()`). W kolumnach powinny być nawy powierzchni, a w wierszach nazwy gatunków. Dlaczego dla niektórych wierszy (gatunków) wyprodukowane zostały wartości `NA`?
   h) Szeroką tabelkę wyeksportować z R do pliku .csv (funkcja `write.table()`). Wartości NA przekształcić w zera (wtedy powiemy R, że te dane nie są brakujące, ale że w danej próbie dany gatunek osiągnął

zerowe pokrycie, czyli w jednej próbie go mogło nie być, a np. w 25 innych próbach notowany był z dużym pokryciem). Pierwszy wiersz przesunąć o jedną kolumnę w prawo. Kolumna A pozostanie wtedy pusta. Należy ją usunąć

i) Szeroką tabelkę załadować ponownie do R
j) Przekształcić tabelkę szeroką w wąską
k) Po załadowaniu do R pliku `cechy.all.csv` do tabelki wąskiej z podpunktu i) dokleić kolumny zawierające wysokość pędów (`canopy_height`) masę liści (`leaf_mass`), rozmiar liści (`leaf_size`) i masę nasion (`seed_mass`). W tym celu należy wykorzystać funkcję `left_join()`. Przed zaimplementowaniem funkcji `left_join()` należy z wczytanej tabeli stworzyć obiekt, zawierający kolumnę z gatunkiem oraz czterema wyżej wymienionymi cechami i nazwać go np. `nasiona.liscie`. link: [https://github.com/mkdyderski/BSS/blob/BSS2019/datasety/cechy.all.csv]. Możesz również ściągnąć go do R za pomocą funkcji `read.csv()`:

```
cechy.all<-read.csv('https://raw.githubusercontent.com/mkdyderski/BSS/BSS2019/datasety/cechy.all.csv',
                    sep=';')
```

l) Korzystając z pakietu `dplyr` dla każdego gatunku z obiektu `nasiona.liscie` obliczyć stosunek powierzchni liści (`leaf_size`) do masy liści (`leaf_mass`) oraz stosunek powierzchni liści (`leaf_mass`) do wysokości pędów (`canopy_height`). Dlaczego dla niektórych gatunków wyprodukowane zostały wartości `NA`?

3. Plik `meteo.csv` (rozdzielony przecinkami – ponieważ R nie czyta przecinków jako znaków decymalnych, trzeba to zdefiniować przy wczytywaniu danych argumentem `dec=','`) zawiera pomiary dobowej temperatury powietrza na Kasprowym Wierchu w latach 1950-2015. link: [https://github.com/mkdyderski/BSS/blob/BSS2019/datasety/lichenes1.csv]. Możesz również ściągnąć go do R za pomocą funkcji `meteo.csv`:

```
meteo<-read.csv('https://raw.githubusercontent.com/mkdyderski/BSS/BSS2019/datasety/meteo.csv',
                    sep=';')
```

Po zaimportowaniu danych do R, przy użyciu pakietu `dplyr` obliczyć liczbę dni ze średnią dobową temperaturą powietrza mniejszą niż 10 stopni Celsjusza w każdym roku monitoringu temperatury oraz obliczyć średnią roczną temperaturę powietrza biorąc pod uwagę tylko temperatury dobowe niższe niż 10 stopni.

Propozycje do pracy z własnym zbiorem danych

4. Wczytaj *własny zbiór danych* i sprawdź poprawność wczytanych danych
5. Spróbuj obliczyć podstawowe statystyki na zmiennych które badasz. Wypróbuj użycie niektórych funkcji.
6. Skonsultuj z prowadzącymi konieczność przekształceń danych - czy będzie trzeba zmienić format danych?
7. Jeśli część danych wymaga złączenia to teraz jest na to najlepszy czas. Nawet jeśli dane mają się zmienić, warto przygotować sobie kod który pozwoli łączyć tabele wg określonego klucza już teraz.
8. Policz średnie wartości i miary dyspersji (SD lub SE, szczegóły poniżej) dla grup w ramach swojego zbioru danych. Mając średnie wartości i miary dyspersji możesz zastanawiać się nad różnicami pomiędzy grupami.

wariancja, SD i SE: wariancja to SD^2, SD jest tutaj estymatorem (przybliżeniem) dyspersji danych SE to SD/sqrt(), czyli jest to SD dzielone przez pierwiastek kwadratowy z liczby prób. W bazowym R nie ma funkcji na SE, więc można ją napisac samemu i wrzucić do konsoli. Funkcje w R to również obiekty - jeśli jakiegoś nie ma, możesz stworzyć go samemu:

```
se<-function(x)sd(x,na.rm=T)/sqrt(length(x[!is.na(x)]))
```