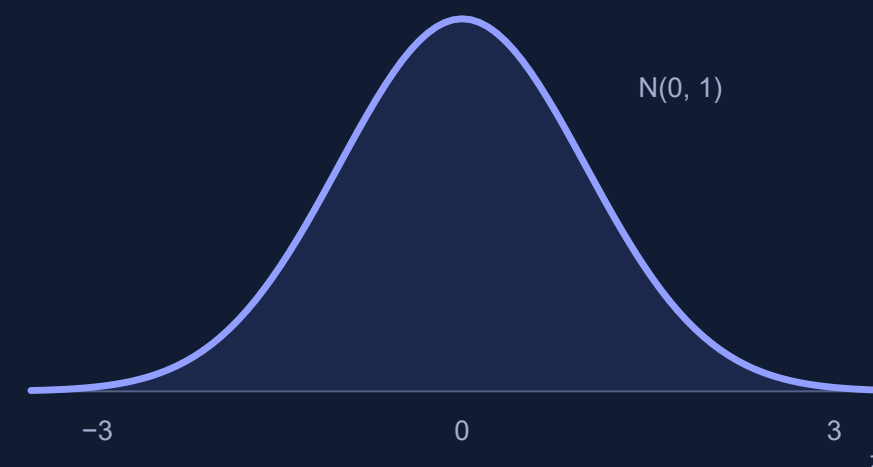


Statystyka I z R

Organizacja zajęć · Wprowadzenie do R

Bartosz Maćkiewicz



Plan zajęć

R jako język programowania i praca z danymi.

1. Podstawowe typy danych (wektory typu numeric, character, logical, integer)
2. Zaawansowane typy danych (ramki danych i listy)
3. Struktury kontrolne (konstrukcje warunkowe, pętle)
4. Funkcje `lapply`, `tapply`, `sapply`
5. Podstawy grafiki w R
6. Wprowadzenie do tidyverse

Plan zajęć

Podstawy statystyki inferencyjnej z R

1. Rozkłady teoretyczne i symulacje
2. Testowanie statystyczne hipotez
3. Test chi-kwadrat
4. Test t-Studenta i przedziały ufności
5. Wielkość efektu i moc testów statystycznych
6. Wstęp do testów nieparametrycznych
7. Korelacja i regresja
8. Regresja liniowa
9. Analiza wariancji

Plan zajęć

Analiza danych eksperymentalnych z R

1. Wprowadzenie do bootstrappingu i innych technik resamplingowych jako alternatywy dla testów nieparametrycznych
2. Podstawy ogólnego modelu liniowego
3. Zmienne nominalne w ogólnym modelu liniowym
4. Interakcje w modelu liniowym
5. Analiza wariancji jako szczególny przypadek modelu liniowego
6. Interakcje między zmiennymi nominalnymi i analiza wariancji
7. Analiza wariancji dla powtórzonych pomiarów
8. Podstawy regresji logistycznej

Tryb pracy na zajęciach

- Obecność na zajęciach obowiązkowa
- Zajęcia - krótki wstęp teoretyczny (jego długość zależy od Państwa przygotowania) i praca z komputerami.
- Prace domowe (około 2-3 godziny tygodniowo, niektórzy potrzebują więcej)
- Prace domowe do wyboru (Większe Prace Domowe, przynajmniej 2 punkty do końca zajęć).
 - Czasami w toku semestru może pojawić się dodatkowe zadanie, które będzie liczyć się jako Większa Praca Domowa
- Czasami coś do przeczytania (według potrzeb)
- Dwa kolokwia w ciągu semestru
 - Kolokwium I: podstawy programowania w R
 - Kolokwium II: statystyczne testowanie hipotez w R

Warunki zaliczenia

- Robienie zadań domowych obowiązkowych + 2 punkty z “większych” prac domowych.
- Zaliczenie (60%) dwóch kolokwiów w semestrze.
- Staramy się nie spóźniać z oddawaniem prac domowych, ale jak ktoś odeśle prace domowe spóźniony to lepiej, niż gdyby miał nigdy nie odeśłać.
- Każdy uczestnik kursu ma prawo do **dwukrotnego tygodniowego spóźnienia bez obniżenia punktacji** w semestrze bez żadnych dodatkowych pytań
- W wyjątkowych sytuacjach można negocjować warunki zaliczenia.

Forma przesyłania prac

- Prace domowe oddajemy na platformie zajęć: źródło Quarto ([.qmd](#)) wraz z wyrenderowanym plikiem HTML.
- Na platformie znajduje się krótkie wprowadzenie do Quarto
- Arkusze mają się renderować, a wyniki mają być we właściwych miejscach. Nienegocjowalne, ale przez pierwsze tygodnie będę Państwa korygował.

Materiały

- Materiały znajdują się na platformie.
- Będę starał się wrzucać dość dużo materiałów i uzupełniać je na bieżąco.
- W razie potrzeb będę aktualizował materiały na stronie.

Korzystanie z narzędzi AI

- **ChatGPT trywializuje znaczną część zadań domowych**, co sprawia, że łatwo jest „ominać” proces nauki.
- **Używanie ChatGPT do zadań domowych jest surowo zabronione**, ponieważ:
 - Utrudnia naukę początkującym, którzy muszą opanować podstawy języka R i statystyki.
 - Uniemożliwia ocenę rzeczywistego zrozumienia i umiejętności studentów, co jest kluczowe na poziomie uniwersyteckim.
 - Sprawdzanie zadań domowych wykonanych przez ChatGPT jest dehumanizujące dla wykładowcy.

Konsekwencje naruszenia zasad

- **Natychmiastowe niezaliczenie przedmiotu** za korzystanie z ChatGPT lub podobnych narzędzi AI.
- **Powiadomienie instytucji akademickich** o naruszeniu zasad uczciwości akademickiej.

Weryfikacja użycia LLMów (system Zofii)

- Nie istnieje stuprocentowo skuteczna metoda wykrywania użycia dużych modeli językowych. Nie oznacza to, że nie ma pewnych poszlak, które wprawne oko jest w stanie wychwycić.
- W przypadku podejrzenia o wykorzystanie LLMów przy pracach domowych osoba studencka zostanie po zajęciach poproszona na rozmowę o napisanym przez siebie kodzie. Jeżeli rozmowa będzie wskazywała na niezrozumienie wysłanego przez siebie zadania, to taka osoba dostaje **czervoną kartkę**.
- Dwie czerwone kartki stanowią dowód wykorzystania LLMów do wykonania prac domowych.

Uzasadnienie polityki

- Duże modele językowe (LLM) jak ChatGPT mogą doskonale radzić sobie z takimi zadaniami jak pisanie widgetów w JavaScript lub centrowanie DIV-ów, ale często zawodzą przy bardziej złożonych i niestandardowych zadaniach.
- **Rozwój krytycznych umiejętności jest zahamowany** przez poleganie na AI jako co-pilocie, w tym:
 - Czytanie dokumentacji.
 - Debugowanie.
 - Interpretacja wyników złożonych analiz.
- **Korzystanie z LLM uniemożliwia nabycie tych kluczowych umiejętności**, zwłaszcza początkującym, którzy dopiero uczą się programowania.

Narzędzia AI — nasza umowa

- Unikaj korzystania z ChatGPT jako co-pilota w swojej drodze edukacyjnej.
- Zdaję sobie sprawę, że LLM mogą być użyteczne w wielu aspektach życia, w tym programowania, **pod warunkiem, że ktoś rozumie podstawy i ma praktyczne umiejętności.**

Python > R? R > Python?

Na Githubie (www.github.com/PanBartosz/semexp) i Youtubie (Bartosz Maćkiewicz) można znaleźć coś w rodzaju nieoficjalnego kursu ze Statystyki I z Pythonem w formie krótkich filmików i notebooków, które będą przekładały przykłady, które robiliśmy w R na Pythona z użyciem takich pakietów jak `numpy`, `pandas`, `scipy`, `statsmodels`.

Dlaczego R?

R jest stworzonym przez Rossa Ihakę i Roberta Gentlemana językiem programowania oraz środowiskiem programistycznym służącym do obliczeń statystycznych. Powstał jako implementacja stworzonego w laboratoriach Bella (UNIX, C, itd.) języka programowania S stworzonego przez Johna Chambersa, Ricka Beckera i Allana Wilksa.

Obecnie rozwijany jest przez *R Development Core Team*.

Zalety

- jest darmowy w przeciwieństwie do wielu innych pakietów statystycznych takich jak SPSS, SAS czy Stata
- jest wolnym oprogramowaniem (kod źródłowy na licencji GNU GPL, sam projekt jest projektem GNU)
- jest ciągle rozwijany
 - ciągły rozwój biblioteki standardowej i olbrzymia liczba pakietów do każdego rodzaju analizy (ale też uczenie maszynowe itp.)
- jest przetestowany
- praktycznie nieograniczone możliwości tworzenia grafiki
- coraz więcej możliwości wykraczania poza same obliczenia (Web, interfejsy graficzne, interaktywne aplikacje, *reproducible research*)
- jest pełnoprawnym językiem programowania

Wady

- jest językiem programowania dla statystyków
 - niezbyt rozbudowane możliwości tworzenia złożonych aplikacji graficznych/webowych (ale sq!)
 - niektóre decyzje dotyczące języka kontrowersyjne
- jest językiem programowania - brak graficznego interfejsu (ale: istnieją nakładki takie jak JASP)
- jest językiem interpretowanym a nie kompilowanym, więc teoretycznie aplikacja w C/C++ może być szybsza

R vs Python

- oba są językami interpretowanymi*, mogą być więc wolne
 - większość operacji numerycznych w R jest zaimplementowana w C/Fortranie, więc będzie szybszy niż czysty Python
 - oba języki mają możliwość pisania rozszerzeń w C, jeżeli wydajność to w Pythonie można użyć `numpy` lub nawet `numba`
- oba są typowane dynamicznie
- podstawowe typy danych w R są wektorowe i operacje na nich są wektoryzowane
- w Pythonie biblioteki mają odseparowaną przestrzeń nazw, w R biblioteki ładują często swoje funkcje do globalnego *namespace*
 - funkcje z zewnętrznych bibliotek mają niesamowite kombinacje wielkich liter i nazw, by odróżnić się od biblioteki standardowej i innych bibliotek

R vs Python

W R możemy wykonać mnożenie dwóch wektorów przez siebie. Jeśli są innej długości, R będzie powtarzał krótszy (nie zawsze to jest super)

```
x = c(1,2,3,4,5,6)
y = c(2,2,3,3)
z = x * y
print(z)
```

```
[1]  2  4  9 12 10 12
```

W Pythonie nie da się przemnożyć dwóch list przez siebie.

```
x = [1,2,3,4,5,6]
y = [2,2,3,3]
z = x * y

print(z)
```

R vs Python

Można jednak użyć pakietu `numpy` i wykonywać operacje na wektorach.

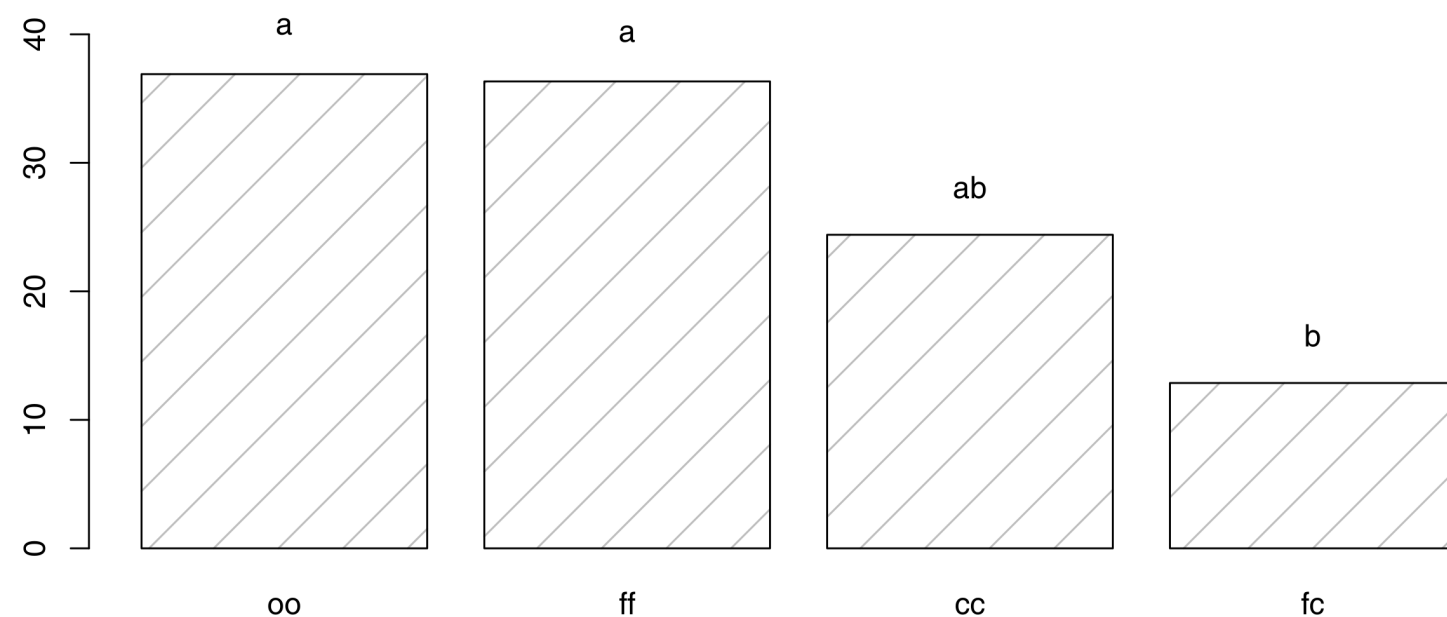
```
import numpy as np
x = np.array([1,2,3,4,5,6])
y = np.array([2,2,3,3])
y = np.concatenate([y,y])[:6]
z = y * x
print(z)
```

```
[ 2  4  9 12 10 12]
```

R vs Python

`LSD.test` oraz `bar.group` są funkcjami, które do globalnej przestrzeni nazw dodał pakiet `agricolae`

```
library(agricolae)
data(sweetpotato)
model<-aov(yield~virus, data=sweetpotato)
out = LSD.test(model,"virus", p.adj="bonferroni")
bar.group(out$groups,ylim=c(0,45),density=4)
```



Jak znaleźć odpowiedzi na nurtujące nas pytania?

- Strona kursu
- Google
- StackOverflow
- podręcznik
- dokumentacja z CRAN (*Comprehensive R Archive Network*)
- funkcja ? ([help](#))

Jak pracować z R?

- REPL (*read-eval-print loop*)
 - interaktywna praca z R
 - szybkie analizy, analiza eksploracyjna (deskryptywna), testowanie
- skrypty R (lub dokumenty Quarto) * traktowane jako notatnik * *pipeline* analizy danych



RStudio

Konsola, edytor kodu i podgląd wyników w jednym miejscu.

logical

Wartości logiczne prawdy i fałszu - TRUE i FALSE

```
piec_falszow = c(FALSE, FALSE, FALSE, FALSE, FALSE)  
print(piec_falszow)
```

```
[1] FALSE FALSE FALSE FALSE FALSE
```

integer

Liczby całkowite

```
liczby_od_jeden_do_piec = as.integer(c(1,2,3,4,5))  
print(liczby_od_jeden_do_piec)
```

```
[1] 1 2 3 4 5
```

numeric

Liczby rzeczywiste

```
prawie_trzy = c(2.95, 2.96, 2.9998, 2.987)
prawie_trzy # Można nie dodawać instrukcji print
```

```
[1] 2.9500 2.9600 2.9998 2.9870
```

character

Ciągi znaków

```
pieciu_mezczyzn = c('M', 'M', 'M', 'M', 'M')  
pieciu_mezczyzn
```

```
[1] "M" "M" "M" "M" "M"
```

factor

W zasadzie są wektorami ciągów znaków, przechowują jednak dodatkowe informacje.

```
table(pieciu_mezczyzn)
## pieciu_mezczyzn
## M
## 5
zadnej_kobiety = factor(pieciu_mezczyzn, levels = c('K', 'M'),
                        ordered = TRUE )
zadnej_kobiety
## [1] M M M M M
## Levels: K < M
table(zadnej_kobiety)
## zadnej_kobiety
## K M
## 0 5
```