Inteligencja obliczeniowa

Laboratorium 3: Czyszczenie, preprocessing i zmniejszanie wymiarowości danych.

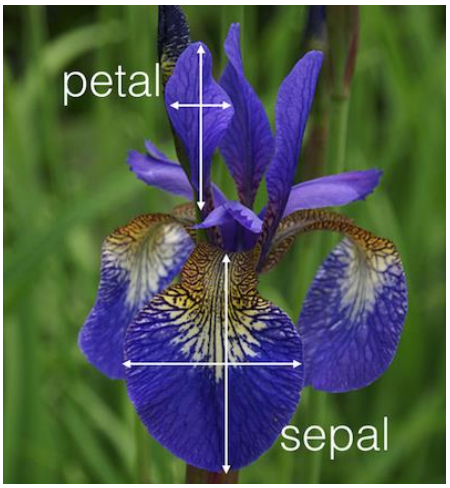
Tematem nadchodzących laboratoriów jest m.in. badanie trzech gatunków irysów.

Iris setosa

Iris versicolor

Iris virginica

Każdemu z trzech gatunków mierzono długość i szerokość płatka „petal" i działki kielicha kwiatu „sepal" (to te większe płatki skierowane w dół).



Mamy więc 4 atrybuty o wartościach rzeczywistych. Stworzono małą bazę danych. Dzisiaj zajmiemy się wstępną obróbką tych danych.

Pozyskane dane mogą nie nadawać się do badania i konieczne jest ich czyszczenie (data cleansing/cleaning) i obróbka. Z reguły najpierw trzeba naprawić bazę danych (znaleźć i usunąć błędy w jej strukturze, złe rekordy, złe dane lub ich zakres czy typ). Następnie mając działającą bazę, należy sprawić by była spójna i logiczna (np. zająć się brakującymi danymi).

W poniższym zadaniu będziemy korzystać z przystępnie napisanego samouczka o czyszczeniu danych dostępnego pod adresem:
https://cran.r-project.org/doc/contrib/de_Jonge+van_der_Loo-Introduction_to_data_cleaning_with_R.pdf
Zapoznaj się zwłaszcza z rozdziałem 3.1.4.

Zadanie 1 „Szukanie błędów"
Wczytaj bazę z irysami dirty_iris.csv (załączone). Następnie wykonaj polecenia:

a) Załaduj bazę danych do R.
```
dirty.iris <- read.csv("dirty_iris.csv", header=TRUE, sep=",")
```
Wyświetl ją i przyjrzyj się rekordom. Sporo rekordów ma brakujące dane (NA = Not Avaialable). Policz ile ze wszystkich 150 rekordów jest pełnych (ma wszystkie dane). Wykorzystaj do tego funkcje:
nrow – liczenie wierszy w tabeli
subset – filtrowanie tabeli
is.finite – sprawdzanie, które dane są liczbami skończonymi
Poprawna odpowiedź: 95.

b) Chcemy stworzyć zestaw reguł, które sprawdzą czy tabela z irysami jest poprawna.

Wykorzystamy do tego paczkę editrules (patrz: https://cran.r-project.org/doc/contrib/de_Jonge+van_der_Loo-Introduction_to_data_cleaning_with_R.pdf strona 35, 36). Zainstaluj i załaduj tę paczkę i dodaj regułę, że długość działki kielicha nie może być dłuższa niż 30 cm:

```
install.packages("editrules")
library(editrules)
E <- editset(c("Sepal.Length <= 30"))
E
```

Następnie sprawdzamy ile wierszy tabeli nie spełnia tej reguły.

```
ve <- violatedEdits(E, dirty.iris)
ve
```

Można wyświetlić też :

```
summary(ve)
plot(ve)
```

c) Popraw editset E, tak aby miał dodatkowo następujące reguły (możesz je wpisać ręcznie w komendzie, lub wczytać z pliku komendą editfile).

- Ostatnia kolumna zawiera tylko wartości: setosa, versicolor, virginica.
- Wszystkie numeryczne wartości muszą być dodatnie.
- Petal.Length musi być minimum dwa razy większe niż Petal.Width
- Sepal jest zawsze dłuższy niż Petal.

d) Sprawdź za pomocą komendy violatedEdits, ile każda z reguł została złamana. Ile danych jest całkowicie poprawnych? Które irysy mają za długie płatki względem ich szerokości?

Zadanie 2 „Wymazanie błędów"

Wykryliśmy błędy, a teraz pora je poprawić. Jeszcze raz wróć do samouczka https://cran.r-project.org/doc/contrib/de_Jonge+van_der_Loo-Introduction_to_data_cleaning_with_R.pdf
i przejdź do rozdziału 3.2.

Korzystając z paczki deducorrect (opisana w podrozdziale 3.2.1).

- Zamień niedodatnie wartości z Petal.Width na wartość NA korzystając z komendy correctWithRules.
- Zastąp wszystkie inne niepoprawne dane wykryte w zadaniu 1 etykietką NA.

Uwaga 1: Operator %in%, który działał w editset, nie będzie działał w tym zadaniu.
Uwaga 2: Sprawdzając czy Petal.Width jest mniejsze równe 0, należy upewnić się czy Petal.Width jest w ogóle liczbą (inaczej wyskoczą nam błędy).

Zadanie 3 „Uzupełnienie braków"

Pozbyliśmy się błędnych danych, ale zostaliśmy z etykietkami NA. Byłoby dobrze, gdybyśmy w ich miejscu mieli liczby. Bezsensowne byłoby wpisywanie jakichkolwiek liczb, ale jest kilka technik, które przynajmniej w przybliżeniu uzupełniają je dobrymi wartościami. Przetestujemy dwie z nich.

a) Wszystkie puste dane z danej kolumny zastąp wartością średnią z kolumny (rozdział 3.3.1). Zastosuj tę technikę i zapisz wyniki w tabeli clean.iris.mean.

b) Najbliżsi sąsiedzi (k-Nearest Neighbor) – rozdział 3.3.3. Gdy irys ma brakujące dane, szukamy k najbardziej podobnych irysów do niego, mających wartości najbardziej zbliżone. Na ich podstawie

wyliczana jest brakująca wartość.
Zastosuj tę technikę i zapisz wyniki w tabeli clean.iris.knn.

Zadanie 4 „Preprocessing"
Gdy dane są naprawione można je jeszcze odpowiednio skalować na dalsze potrzeby. Przygotowanie danych dla różnych algorytmów (które poznamy wkrótce) nazywamy preprocessingiem.

a) Wpisz iris w konsoli R. Powinna wyświetlić się tabela z bazą danych irysów. Nie zawiera ona błędów, z którymi musieliśmy męczyć się w poprzednich zadaniach.

b) Wszystkie cztery numeryczne kolumny zlogarytmuj (funkcja log). Wynikową tabelę zapisz pod nazwą iris.log.
Logarytmowanie ma sens gdy badamy nie wartości bezwzględne, a raczej zależności między wartościami i powiązania. Szerzej omówione jest to w załączonym do zajęć pliku log.pdf. Warto się w niego wczytać.

c) Znamy już funkcję standaryzującą. Kompresuje ona wartości – umieszczając je w okolicach zera, tak że średnio są oddalone od zera o 1. Taki preprocessing nie tylko umożlwia lepsze działanie dla wielu algorytmów, ale też często lepiej wygląda na wykresach (zwłaszcza, jak musimy porównywać kilka zmiennych o różnych zakresach).

Funkcja standaryzująca dostepna jest w R pod nazwą scale. Dokonaj standaryzacji czterech kolumn numerycznych tabeli iris.log. Tabelę zapisz pod nazwą iris.log.scale.

Zadanie 5 „PCA"
Zajmiemy się teraz analizą głównych składowych dla tabeli dla tabeli `iris.log.scale`.
Analiza głównych składowych to pewna metoda na zmniejszanie wymiarowości bazy danych. Chcemy 4 parametry kwiatów zamienić na mniejszą ilość parametrów, które nadal będą dobrze charakteryzowały kwiaty. Gdy zredukujemy ilość atrybutów np. do dwóch (dwie główne składowe), to nie dość, że baza danych będzie „odchudzona" to w dodatku będzie można kwiaty umieścić na wykresie 2D, gdzie wyodrębnią się pewne grupy.

a) Usuń kolumnę z nazwami irysów z `iris.log.scale`.

b) Dla tak zmienionej tabeli, skorzystaj z funkcji `prcomp`, by obliczyć główne składowe i rezultat zapisz pod nazwą `iris.pca`.

c) Wyświetl za pomocą komendy `iris.pca` lub `print(iris.pca)` dane dotyczące głównych składowych. Odpowiedz na pytania:

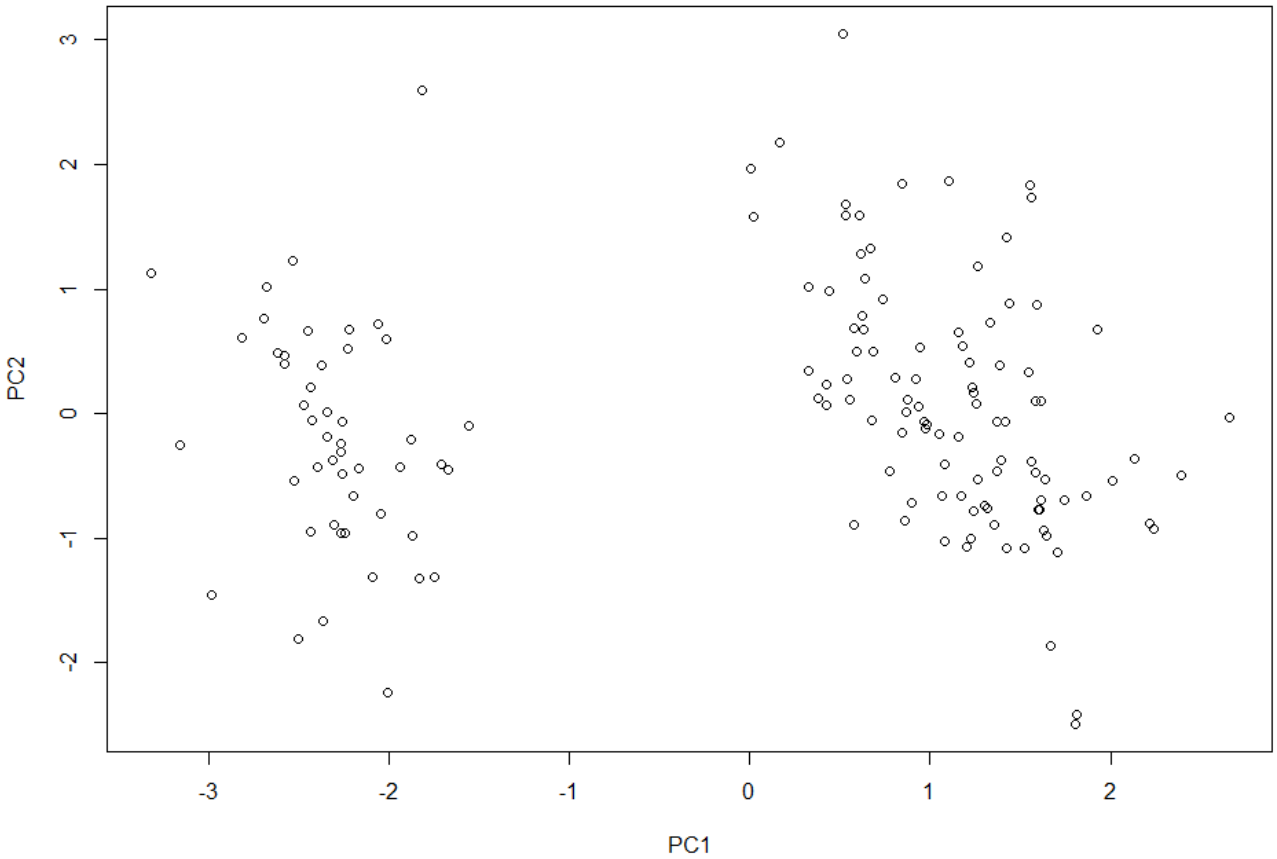
- Jakie odchylenia standardowe mają główne składowe PC1, PC2, PC3, PC4? Jeśli chcemy zostawić tylko dwie główne składowe o największym odchyleniu standardowym, to które to będą?
- Co zwracają komendy `iris.pca[1]` i `iris.pca[2]`? Co zwróci komenda `iris.pca[2]$rotation`?
- Macierz rotacji podaje przepis na współczynniki liniowe do tworzenia składowych PC1,...,PC4. Przykładowo, zeby stworzyć PC1 dla jednego z irysów, trzeba wziąć wszystkie jego pomiary płatków, pomnożyć przez odpowiednie współczynniki z tej macierzy, dodać i otrzymamy PC1 (innymi słowy: pomnożyć skalarnie wektor z danymi z płatkami i pierwszą kolumną iris.pca[2]$rotation).

c) Jaką tabelę zwraca komenda `predict(iris.pca)`?
Zapisz to pod
`iris.predict <- predict(iris.pca)`

Jest to baza irysów po zadziałaniu macierzą rotation na naszą bazę iris.slog.scale. Ponieważ chcemy zachować tylko dwie najlepsze wartości PC (które to?) to wytnij niepotrzebne dwie kolumny z iris.predict. Doklej z powrotem kolumnę z nazwami irysów. Będą zatem trzy kolumny w bazie danych.

Zadanie 6 „Wykres PCA"
Naniesiemy teraz dane na wykres. Osie x i y będą odpowiadały głównym składowym PC1, PC2. Każdy kwiat będzie punktem na wykresie. Zrób tak, aby każdy rodzaj kwiatu (setosa, versicolor, virginica) miał **inny kolor** na wykresie. Dodaj **legendę**.

Poniżej graf jeszcze przed kolorowaniem punktów i dodaniem legendy.



Gdy skończysz tworzyć wykres, sprawdź czy każdy z gatunków ma swoją własną charakterystykę, skupia się w określonym obszarze.

Niedokończone na laboratoriach zadania zrób w domu.