МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЦЕНТРАЛЬНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання лабораторних робіт з навчальної дисципліни «Big Data» для студентів денної та заочної форми навчання за спеціальностями
123 «Комп’ютерна інженерія»,
122 «Комп’ютерні науки»

ЗАТВЕРДЖЕНО
на засіданні кафедри кібербезпеки та програмного забезпечення,
протокол №12 від 19.03.2024 року

КРОПИВНИЦЬКИЙ
2024

Методичні рекомендації до виконання лабораторних робіт з навчальної дисципліни «Big Data» для студентів денної та заочної форми навчання за спеціальностями 123 «Комп’ютерна інженерія», 122 «Комп’ютерні науки» / уклад. В.В. Босько, Л.В. Константинова, Дрєєва Г.М. — Кропивницький: ЦНТУ, 2024. — 73 с.

Укладач: Босько В. В., Константинова Л.В., Дрєєва Г.М.

Рецензенти: Смірнов О. А., д-р техн. наук, професор;
Мелешко Є.В., д-р техн. наук, професор.

Вступ

В методичних рекомендаціях представлено теоретичний і практичний матеріал для роботи з великими даними.

Організація даних є ключовим моментом під час роботи з великою кількістю за об'ємом інформації. Дуже важливо впорядкувати дані таким чином, щоб легко та швидко знаходити потрібну інформацію. Для роботи з великими даними розглядається мова R.

R - це і мова програмування, і програмне забезпечення, що було створене Россом Іакою та Робертом Джентльменом у 1993 р. Популярність R пояснюється неперевершеною гнучкістю для аналізу даних і потужною графікою. R має великий каталог статистичних та графічних методів [1]. Основні архіви вихідного коду зберігаються спеціальною групою, відомою як R Core Team. Команда випускає оновлені версії R порівняно часто.

Теми лабораторних робіт, що розглядаються та оцінювання знань

Дані методичні рекомендації містять у собі матеріал з 8 лабораторних робіт за наступними темами:

Теми лабораторних робіт
1. Знайомство з мовою R
2. Стратегії роботи з великими масивами даних
3. Підготовка вихідних даних
4. Обробка даних. Вибір ознак (FeatureSelection)
5. Обробка даних. Вибір екземплярів (InstanceSelection)
6. Обробка даних. Дискретизація для класифікації (Discretization)
7. Організація розподілених обчислень
8. Практичні завдання з використанням різних інструментів обробки великих даних

Здобувачам потрібно виконати завдання до лабораторних робіт, а також відповісти на запитання, що знаходяться в кінці кожної лабораторної роботи.

Звіт повинен містити хід виконання завдань а також графічні матеріали, що підтверджують виконання цих завдань. Приклад звіту знаходиться у додатку 1.

Максимальну кількість балів студент може одержати у випадку відвідування всіх лекцій, лабораторних занять, виконання і захисту виконаних завдань для самостійної роботи у встановлений термін та проходження контролю вчасно.

У разі виконання й захисту лабораторних робіт після встановленого терміну, одержані бали вираховуються з коефіцієнтом: для самостійної роботи студента -0,3; лабораторної роботи -0,7.

Звіт готується в паперовому варіанті й представляється під час захисту лабораторної роботи. В окремих випадках надсилається електронний варіант звіту за електронною адресою викладачу. Ім’я файлу звіту повинно містити прізвище, ім’я та групу студента, дисципліну та номер лабораторної роботи, що захищається.

Шкала оцінювання: національна та ECTS

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою
		для екзамену, курсової роботи
90 – 100	**A**	відмінно
82-89	**B**	добре
74-81	**C**	
64-73	**D**	задовільно
60-63	**E**	
35-59	**FX**	незадовільно з можливістю повторного складання
0-34	**F**	незадовільно з обов’язковим повторним вивченням дисципліни

Лабораторна робота №1

Тема: Знайомство з мовою R

Мета: Ознайомитись з основними характеристиками мови R, навчитись встановлювати R та працювати в середовищі RStudio

Завдання:

1. Встановити та запустити програму R.
2. Налаштувати робочу директорію за допомогою функції setwd ().
3. Згенерувати вектор X ~ U (0, 1) і побудувати гістограму.
4. Зберегти результати моделювання в текстовий файл (* .txt, * .csv), гистограмму в графічний файл (* .jpg, * .svg) з використанням функцій запису в файл.
5. Ознайомитись з основними структурами даних numeric (), matrix (), data.frame (), list ().
6. Ознайомитись з довідковою системою.
7. Установка і завантаження додаткових пакетів за допомогою функцій install.packages () і library ().

Теоретичні відомості:

R - це вільнопоширюваний програмний продукт, який динамічно розвивається і використовується провідними науковими установами в різних наукових галузях[1].

Щоб встановити і запустити програму R необхідно перейти за відповідним посиланням, вибрать відповідну версію та підтвердити виконання простими діями (https://cran.r-project.org/).

Для виконання завдання можна скористатись IDE для R, наприклад, RStudio.

RStudio - це інтегроване середовище розробки (IDE) для R. Воно містить консоль, редактор підсвічування синтаксису, який підтримує пряме виконання коду, а також інструменти для побудови графіків, історії, налагодження та керування робочим простором.

R – це чутлива до регістру клавіатури інтерпретована мова програмування [2].

#Простий приклад :

Hello<- “Hello world!”

Print hello

#це коментар на всякий випадок ;)

Дані - інформація, оброблена і подана у формалізованому вигляді для подальшої обробки.

Життєвий цикл даних - це послідовність етапів, яку конкретна порція даних проходить від початкового етапу створення або отримання до моменту архівації або видалення. Створення даних, обслуговування даних, синтез даних, використання даних, публікація даних, архівація даних, знищення даних – це **основні етапи** життєвого циклу даних.

Типи даних, що використовується мовою R: вектори, матриці, масиви, таблиці даних і списки (сукупність кількох об’єктів). Типи даних в R бувають числовими (numeric), текстовими (character), логічними (TRUE/FALSE, правда/брехня), комплексними (уявне число) і необробленими (байти) [1][2][3].

Змінні - це такі контейнери, в які можна покласти різні дані й навіть функції. Імена змінних можуть складатись з латинських літер обох регістрів, символів точки. і підкреслення _ , а також цифр. Імена змінних мають починатися з латинських букв.

Оператор присвоювання це символ стрілки <-. Він працює так, що значення виразу в його правій частині надається об'єкту в лівій частині.

Вектор - це список деяких об’єктів (одномірний масив даних). Для створення вектора застосовується функція об’єднання c(). Наприклад:

q <- c(1, 2, 0, 3) # вектор q

Скаляри – це вектори, що складаються з одного елемента (наприклад, f <- 3, використовуються для позначення констант).

Визначити довжину вектору:

length(q)

Матриця (matrix) – це двовимірний масив даних, у якому кожен елемент має однаковий тип і створюють з допомогою функції matrix().

Масиви даних (array) схожі з матрицями, але можуть мати понад два виміри і створюються з допомогою функції array().

Таблиця даних (data frame) – це більш широко використовуваний у порівнянні з матрицею об'єкт, оскільки різні стовпці можуть містити різні типи даних (числовий, текстовий та ін.).

Поточна робоча директорія – це та директорія, де містяться файли даних і куди за замовчуванням зберігаються результати. Функція getwd() дозволяє дізнатися, яка директорія в даний момент є робочою. Призначити робочу директорію - функція setwd().

Щоб отримати уявлення щодо графічних можливостях R, наберіть у командному рядку: demo(graphics).

Статистичні дані для зручності зазвичай зображають за допомогою статистичних таблиць та статистичних графіків. Графічне зображення даних реалізовується досить ефективно, для цього наявна значна кількість різних процедур (представлено в таблиці 1.1).

Базовою функцією роботи з графіками є функція plot().

Таблиця 1.1 [3] – Функції для графічного зображення в R

Графік	Функція	Опис
Графік "ящик з вусами"	`boxplot()`	є зображенням квартилей розподілу неперервної характеристики
Гістограма	`hist()`	один з можливих засобів зобразити розподіл деякої неперервної характеристики. Її можна розглядати як оцінку щільності відповідного розподілу
Графік індивідуальних значень	`stripchart()`, `dotchart()`	застосовують, якщо досліджують дискретну характеристику і дані зосереджені у вузькому діапазоні
Мозаїка	`mosaicplot()`	дозволяє зобразити таблицю спряженості у вигляді діаграми
Стовпчаста діаграма	`barplot()`	застосовують, якщо необхідно зобразити значення кількісних дискретних або якісних характеристик
Кругова діаграма	`pie()`	дозволяє зобразити структуру статистичної сукупності
Емпірична функція розподілу	`ecdf()`	є зображенням емпіричної функції розподілу.
Квантиль–квантиль графік	`qqnorm()`	застосовують, якщо досліджують відповідність сукупності значень нормальному розподілу

Для виконання завдань можна використати такий алгоритм дій:

1. Для встановлення R виконуються наступні дії:

З сайту https://posit.co/download/rstudio-desktop/#download можна завантажити та всановити, як розглядається на рисунку 1.1 та на рисунку 1.2.

Рисунок 1.1 - Завантаження

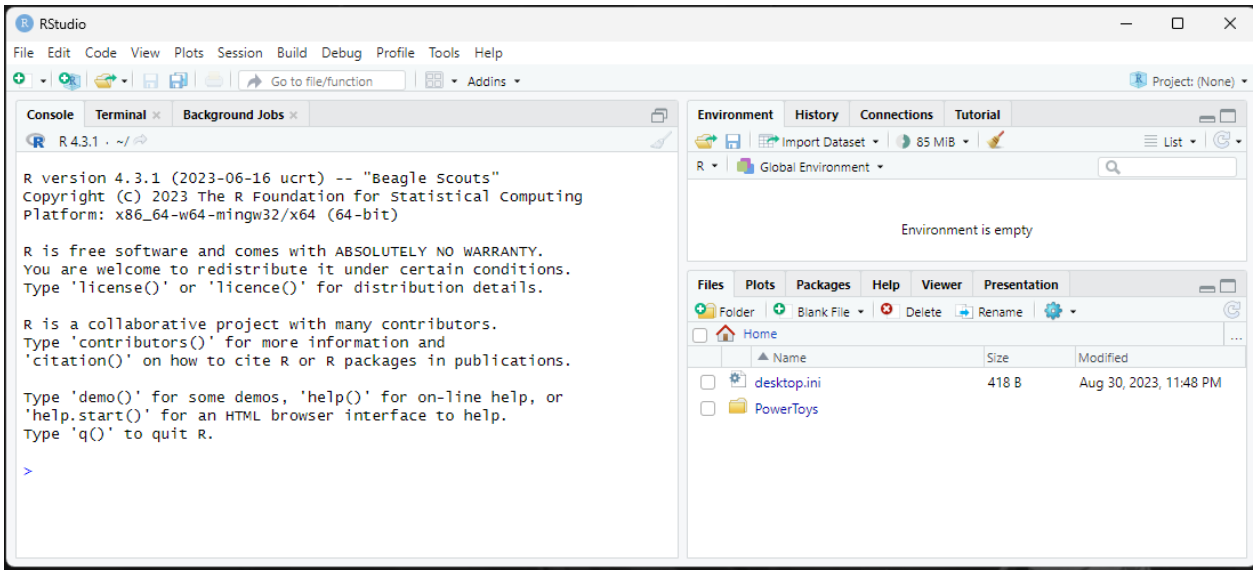


Рисунок 1.2 – Встановлено RStudio

2. Для налаштування робочої директорії за допомогою функції setwd() виконуються дії та послідовності команд, що вказані на рисунку 1.3.

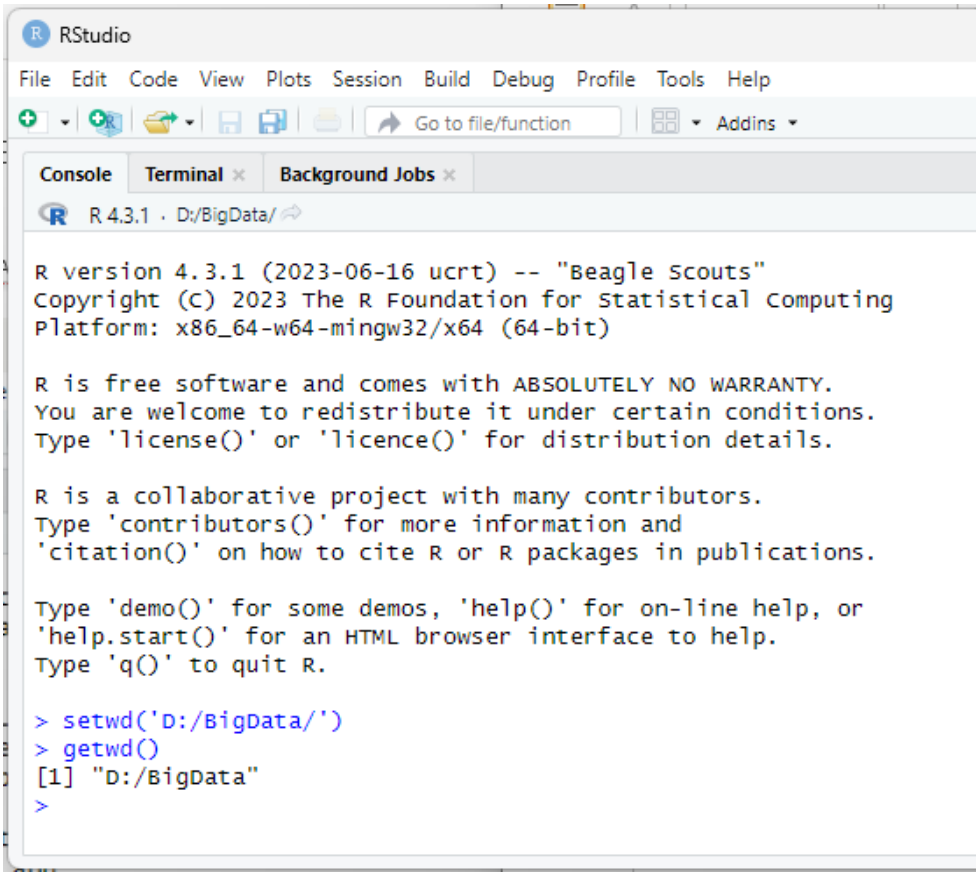


Рисунок 1.3 – Робота в RStudio, налаштування робочої директорії

3. Для того, щоб згенерувати вектор X ~ U (0, 1) і побудувати гістограму виконується наступна послідовність команд, результат видно на рисунку 1.4.

```
>X<- seq(0, 1, len=11)
>hist(X, main = "ГістограмапослідовностіX ~ U(0, 1)",
+      xlab = "Значення", ylab = "Частота", col = "green")
```

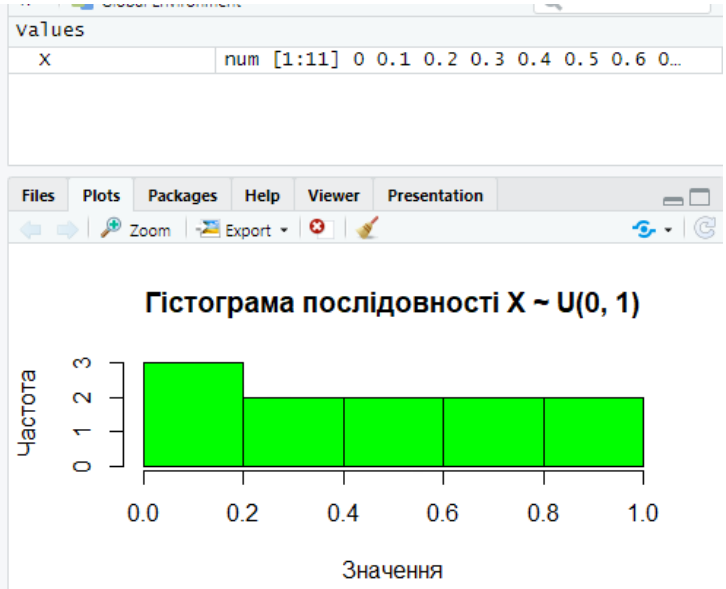


Рисунок 1.4 – Побудова гістограми

4. Зберегти результати моделювання в текстовий файл (* .txt, * .csv), гістограмму в графічний файл (* .jpg, * .svg) з використанням функцій запису в файл можна наступним чином (рисунок 1.5):

Для текстового файлу:

>write.csv(X, "lab1.csv")

Для збереження гістограми в форматі svg:

>svg("histogram.svg")

>hist(X, main = "Гістограма послідовності sX ~ U(0, 1)",

+ xlab = "Значення", ylab = "Частота", col = "green")

>dev.off()

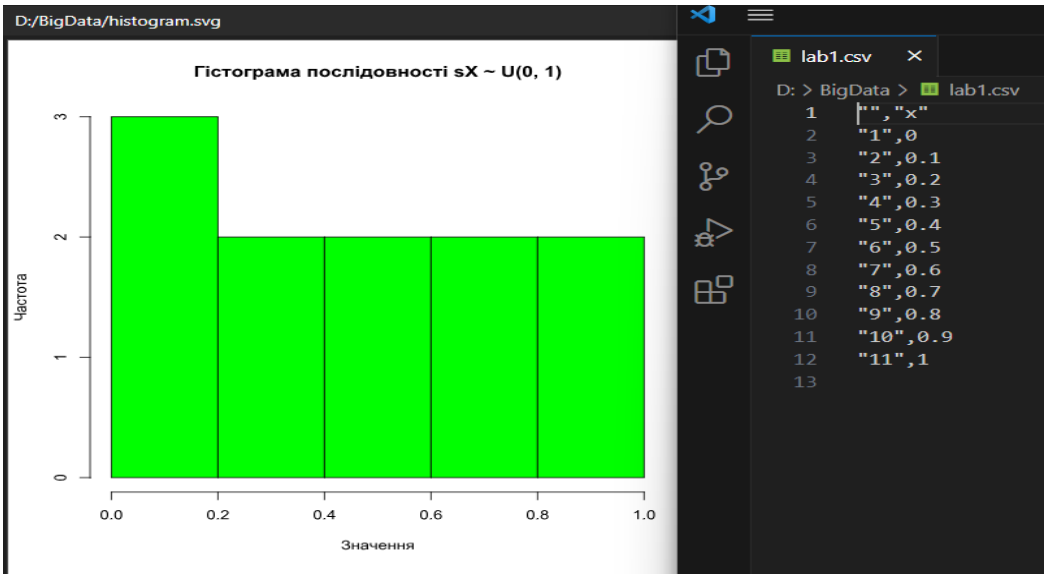


Рисунок 1.5 – Збереження гістограми в форматі svg

5. Для ознайомлення з основними структурами даних: numeric (), matrix (), data.frame (), list () необхідно спочатку підготуватись теоретично потім застосувати у своїй роботі ці структури, наприклад, як вказано далі та на рисунку 1.6.

Numeric() - використовується для створення векторів чисел з плаваючою комою, вказавши їх значення у вигляді послідовності:

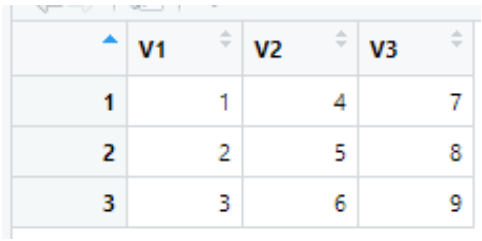
>numeric_vector<- c(1.5, 2.3, 0.8)

```
numeric_vector  num [1:3] 1.5 2.3 0.8
```

Matrix() - використовується для створення матриць. Можливо створити матрицю, вказавши її розмірність та вміст.

>mx <- matrix(1:9, nrow = 3, ncol = 3)

>View(mx



	V1	V2	V3
1	1	4	7
2	2	5	8
3	3	6	9

Рисунок 1.6 – Застосування матриці

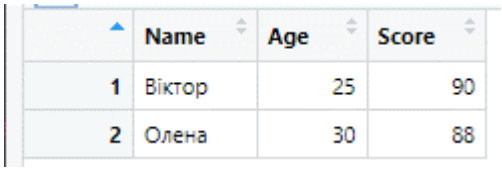
Data.frame() - використовується для створення датафреймів. Тобто таблиці, де кожний стовпець може бути різним типом даних (рисунок 1.7).

>df<- data.frame(Name = c("Віктор", "Олена"),

+ Age = c(25, 30),

+ Score = c(90, 88))

>View(df)



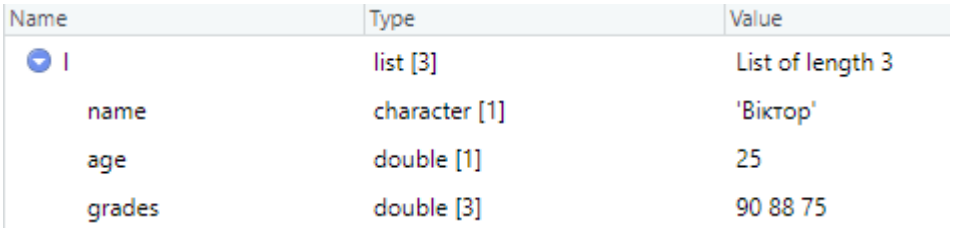
	Name	Age	Score
1	Віктор	25	90
2	Олена	30	88

Рисунок 1.7 – Створення датафрейму

List() - використовується для створення списків, де кожен елемент може бути різним типом даних, також можуть містити в собі вектори, матриці, датафрейми та інші списки [4]. Наприклад наступні рядки і рисунок 1.8:

>l <- list(name = "Віктор", age = 25, grades = c(90, 88, 75))

>View(l)



Name	Type	Value
l	list [3]	List of length 3
name	character [1]	'Віктор'
age	double [1]	25
grades	double [3]	90 88 75

Рисунок 1.8 – Створення списку

6. Ознайомитись з довідковою системою.

Щоб отримати допомогу: встановити курсор на слово і натиснуть F1; перед назвою функції ставимо ? або застосувати функцію help(), наприклад: help.search(“sort”) - здійснити пошук у довідці записів про сортування та переглянути вікно результату або як зображено на рисунку 1.9.

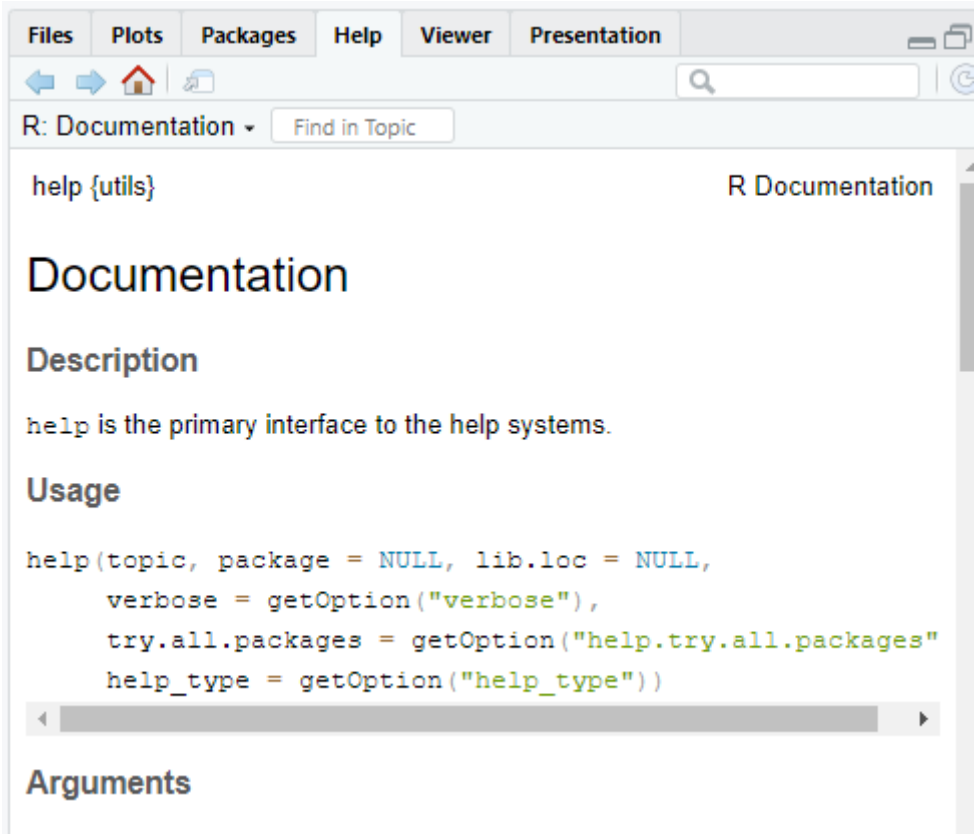


Рисунок 1.9 – Застосування функції help()

7. Установка і завантаження додаткових пакетів за допомогою функцій install.packages () ilibrary ().

Для прикладу встановимо пакет readxl для читання Exel файлів (рисунок 1.10).

```
> install.packages("readxl")
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rto
ols before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/devon/AppData/Local/R/win-library/
4.3'
(as 'lib' is unspecified)
```

```
> library("readxl")
> read_excel("D:/BigData/test.xlsx")
# A tibble: 3 x 2
  Product Price
  <chr>   <dbl>
1 Coffee      5
2 Bread      10
3 Sult       17
```

Рисунок 1.10 – Пакет readxl для читання Exel файлів

Допоміжні джерела інформації:

1. Install R and RStudio on Windows 7 (OpenIntro):

https://www.youtube.com/watch?v=eD07NznguA4

2. R Programming for Beginners | Complete Tutorial | R & RStudio: https://www.youtube.com/watch?v=BvKETZ6kr9Q

3. R Programming Tutorial - Learn the Basics of Statistical Computing: https://www.youtube.com/watch?v=_V8eKsto3Ug

4. Лекція про R: https://rstudio-pubs-static.s3.amazonaws.com/378105_599bcd2892bf46498a6371290149267d.html

5. Курси: https://dev.ua/news/12-kursiv-z-movy-prohramuvannia-r-vid-bazovykh-do-superprosunutykh

6. Англомовна література: https://www.r-project.org/doc/bib/R-books.html

7. Довідка з R: https://tvimc.jimdofree.com/%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D1%87%D0%BD%D1%96-%D0%BC%D0%B0%D1%82%D0%B5%D1%80%D1%96%D0%B0%D0%BB%D0%B8/%D0%B4%D0%BE%D0%B2%D1%96%D0%B4%D0%BA%D0%B0-%D0%B7-r/

Контрольні запитання до лабораторної роботи 1:

1. Що розуміють під поняттям дані?
2. Що представляє собою життєвий цикл даних?
3. Які основні етапи життєвого циклу даних?
4. Що представляє собою мова R?
5. Яким чином оголошують змінні в R та привласнюють їм значення?
6. За допомогою якої функції можна налаштувати робочу директорію?
7. За допомогою якої функції можна перевірити яка встановлена робоча директорія?
8. Що представляє собою вектор?
9. Як визначити кількість елементів вектору?

10. Як побудувати гістограму в R?
11. Як здійснити пошук у довідці записів про сортування?
12. Перелічіть основні структури даних в R?
13. Як здійснити встановлення та завантаження додаткових пакетів?

Лабораторна робота №2

Тема: Стратегії роботи з великими масивами даних

Мета: Ознайомитись з основними стратегіями роботи з великими масивами даних та фреймами за допомогою R

Завдання:

1. Згенерувати великий масив даних і записати в один файл, встановити пакет purrr. Записати масив даних по частинах в кілька файлів. Сформувати репрезентативну вибірку обмеженого розміру.
2. Виконати завантаження даних з використанням різних стратегій. Зробити висновки. Встановити пакети data.table, sqldf, ff.
3. Встановити і завантажити бібліотеки sqldf і nycflights13.
4. Ознакомитись зі структурою набору даних flights. Обчислити кількість спостережень для всіх перевізників carrier в таблиці flights. Відобразити в консолі значення полей dep_time, dep_delay, arr_time, carrier,tailnum з таблиці flights (перші і останні 5 рядків). Обчислити середній час затримки прибуття (mean_arr_delay) і відправлення (mean_dep_delay) для різних перевізників (carrier).
5. Згенерувати data.frame з трьома стовпцями і 100 рядками. Перетворити дані з широкого в довгий формат. Встановити пакет reshape2.

Теоретичні відомості:

Масиви – це об'єкти даних R, які можуть зберігати дані у більш ніж двох вимірах [4].

Масив створюється за допомогою функції array(). Він приймає вектори як вхідні дані та використовує значення у параметрі dim для створення масиву.

Приклад:

```
# Create two vectors
vector01 <- c(4,8,3)
vector02 <- c(11,12,13,14,15,16)

# input to the array
result <- array(c(vector01,vector02),dim = c(3,3,2))
```

print(result)

Стандартним засобом подання табличних даних R є фрейми даних [4].

Фрейм даних являє собою таблицю або двовимірну масивоподібну структуру, в якій кожен стовпець містить значення однієї змінної, а кожен рядок містить один набір значень кожного стовпця.

Стовпці, рядки і осередки є основними структурними елементами фрейму даних або тіббла.

Якщо проаналізувати найбільш типові маніпуляції, які здійснюються над таблицями, їх виявиться небагато: вибір змінних, фільтрація рядків, сортування, обчислення нових стовпців, агрегування, статистики і угруповання. Усі ці завдання можна вирішувати стандартними засобами R . Однак деякі з них досить громіздкі у реалізації (наприклад, угруповання). Також R дозволяє використовувати готові засоби, що справляються з подібними завданнями простим та елегантним шляхом.

Щоб отримати весь список встановлених пакетів можна скористатись функцією: library(). Щоб отримати всі пакети, що завантажені в даний момент: search().

Пакет artifact purrr - це пакет функціональної мови програмування високого рівня, написаний Хедлі Вікхемом. Він може заповнити відсутні частини функціонального програмування R, роблячи програмування більш функціональним.

Data.table - це пакет R, який надає розширену версію data.frames, що є стандартною структурою даних для зберігання даних. Перевага data.table - у суворому лаконічному синтаксисі та найефективніших алгоритмах, що забезпечують дуже серйозний приріст у швидкості роботи з даними [5].

Встановити пакет можливо за допомогою стандартної команди install.packages('…'). Як тільки пакет встановлений, то він відразу готовий до роботи. Потрібно тільки ініціалізувати його перед вживанням. Для цього служить команда library().

Sqldf - це пакет R для виконання операторів SQL на фреймах даних R, оптимізований для зручності [6].

В R можна працювати як з датами, так і з часом. Ми будемо використовувати датасет flights з пакету nycflights13. Наприклад:

```
library(nycflights13)
flights
…
```

Пакет nycflights13 містить дані авіакомпаній про вчасність для всіх рейсів, що відправляються з Нью-Йорка в 2013 році. Також містяться корисні «метадані» про авіакомпанії, аеропорти, погоду та літаки.

За допомогою data.table можливе ефективне переформатування даних.

Пакет reshape2 – це гнучке змінення форми даних: перезавантаження пакета Reshape. Доступне використання функцій для переформатування даних у таблицях data.tables melt (з "широкого" формату в "довгий") і dcast (з "довгого" формату в «широкий»), а також нова розширена функціональність для множинних стовпців. Функції melt та dcast для таблиць data.tables є розширеннями відповідних функцій із пакету reshape2.

Вибірка це набір значень, отриманих в результаті ряду вимірювань. Існує дві загальні характеристики вибірки: центр та розкид. В якості центральної характеристики найчастіше використовуються середнє і медіана, а як розкид стандартне відхилення і квартилі.

Припустимо, що ми маємо вибірку деякого обмеженого обсягу, і вважаємо цю вибірку репрезентативною (тобто вибірка задовільно відображає властивості генеральної сукупності, з якої вона була взята). Ідея бутстреп-методу полягає в тому, що ми можемо вилучити велику кількість випадкових вибірок із цієї сукупності для обрахунку шуканого параметру (або параметрів). Очевидно, що завдяки випадковому процесу формування цих нових вибірок, буде спостерігатися певна варіація значень оцінюваного параметру. Іншими словами, ми отримаємо деякий розподіл значень цього параметру. Розрахувавши стандартне відхилення розподілу, ми отримаємо оцінку стандартної помилки параметру, яка за великої кількості спостережень асимптотично наближатиметься до істинної стандартної помилки. Аналогічно отримують оцінки границь довірчого інтервалу [7].

Для виконання завдань можна використати такий алгоритм дій:

1. Для першого завдання можна виконувати наступні дії:

Для початку встановимо пакет «purrr»:

```
>setwd("D:/BigData/")
>install.packages("purrr")
>library(purrr)
```

Створимо великий масив даних:

```
>large_data<- runif(1000)
```

Запишемо в один файл

```
write.table(large_data, file = "large_data.txt", col.names = FALSE, row.names = FALSE)
```

Наступним етапом буде розбиття масиву на частини (на 10 частин по 100 елементів).

```
>split_data<- split(large_data, rep(1:10, each = 100))
```

Запишемо кожну частину в окремий файл з нумерацією:

```
>for (i in seq_along(split_data)) {
+     filename <- paste0("part", i, ".txt")
+ write.table(split_data[[i]], file = filename, col.names = FALSE, row.names = FALSE)
+ }
```

Отримаємо набір текстових файлів в теці нашої робочої директорії.

Сформуєм репрезентативну вибірку обмеженого розміру, а саме 40 елементів.

```
>sample_data<- sample(large_data, size = 40)
```

Запишемо її у файл:

```
>write.table(sample_data, file = "sample_data.txt", col.names = FALSE, row.names = FALSE)
```

2. Для виконання другого завдання на початку встановимо всі пакети:

```
>install.packages("data.table")
>install.packages("sqldf")
>install.packages("ff")
```

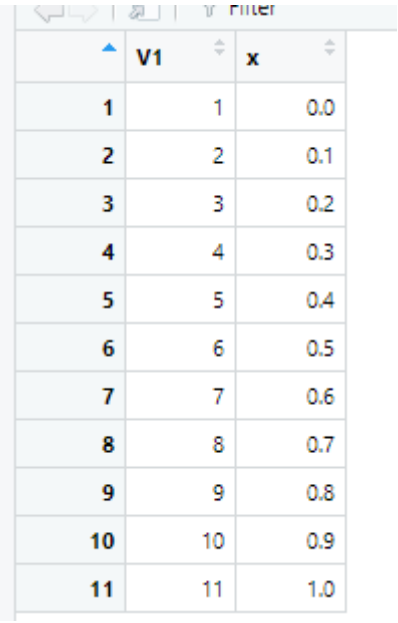
І спробуємо кожну по черзі

```
>library(data.table)
```

>my_data<- fread("lab1.csv")

>View(my_data)

Отримали таблицю представлену на рисунку 2.1.



	V1	x
1	1	0.0
2	2	0.1
3	3	0.2
4	4	0.3
5	5	0.4
6	6	0.5
7	7	0.6
8	8	0.7
9	9	0.8
10	10	0.9
11	11	1.0

Рисунок 2.1 – Застосування library(data.table)

SQLDF:

>library(sqldf)

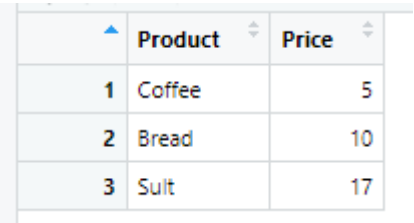
>my_data<- read_excel("D:/BigData/test.xlsx")

Виконаємо запит в стилі sql:

result <- sqldf("SELECT * FROM my_data ")

>View(result)

Отримали таблицю представлену на рисунку 2.2.



	Product	Price
1	Coffee	5
2	Bread	10
3	Suit	17

Рисунок 2.2 – Застосування library(sqldf)

FF:

>library(ff)

>my_data<- read.csv.ffdf(file = "large_data.csv", header = TRUE)

Результат представлено на рисунку 2.3.

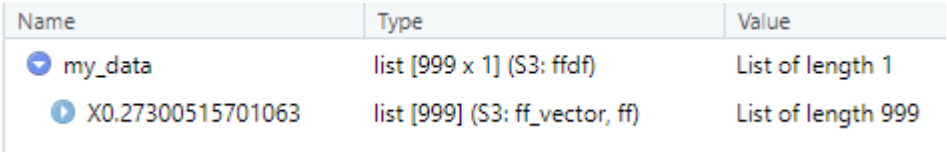


Рисунок 2.3 – Застосування library(ff)

3. Простий приклад встановлення і завантаження бібліотек sqldf і nycflights13. Sqldf встановлювалась в попередньому завдані.

Встановимо nycflights13:

>install.packages("nycflights13")

Та завантажимо:

>library(nycflights13)

4. Ознайомитись зі структурою набору даних flights. Обчислити кількість спостережень для всіх перевізників carrier в таблиці flights. Відобразити в консолі значення полів dep_time, dep_delay, arr_time, carrier, tailnum з таблиці flights (перші і останні 5 рядків). Обчислити середній час затримки прибуття (mean_arr_delay) і відправлення (mean_dep_delay) для різних перевізників (carrier):

Завантажується структура набору даних flights:

Результат представлено на рисунку 2.4.

```
> str(flights)
tibble [336,776 × 19] (S3: tbl_df/tbl/data.frame)
 $ year          : int [1:336776] 2013 2013 2013 2013 2013 2013 2013 2013 2013 2013 ...
 $ month         : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...
 $ day           : int [1:336776] 1 1 1 1 1 1 1 1 1 1 ...
 $ dep_time      : int [1:336776] 517 533 542 544 554 554 555 557 557 558 ...
 $ sched_dep_time: int [1:336776] 515 529 540 545 600 558 600 600 600 600 ...
 $ dep_delay     : num [1:336776] 2 4 2 -1 -6 -4 -5 -3 -3 -2 ...
 $ arr_time      : int [1:336776] 830 850 923 1004 812 740 913 709 838 753 ...
 $ sched_arr_time: int [1:336776] 819 830 850 1022 837 728 854 723 846 745 ...
```

Рисунок 2.4 – Структура набору даних flights

Обчислюється кількість спостережень для всіх перевізників:

>table(flights$carrier)

Результат представлено на рисунку 2.5.

```
   9E    AA    AS    B6    DL    EV    F9    FL    HA    MQ    OO    UA    US    VX    WN    YV
18460 32729   714 54635 48110 54173   685  3260   342 26397    32 58665 20536  5162 12275   601
```

Рисунок 2.5 – Кількість спостережень

Відобразимо значення полів dep_time, dep_delay, arr_time, carrier і tailnum для перших і останніх 5 рядків таблиці flights.

Перші 5(рисунок 2.6):

>head(flights[, c("dep_time", "dep_delay", "arr_time", "carrier", "tailnum")], 5)

```
# A tibble: 5 × 5
  dep_time dep_delay arr_time carrier tailnum
     <int>     <dbl>    <int> <chr>   <chr>
1      517         2      830 UA      N14228
2      533         4      850 UA      N24211
3      542         2      923 AA      N619AA
4      544        -1     1004 B6      N804JB
5      554        -6      812 DL      N668DN
```

Рисунок 2.6 – Перші 5 рядків таблиці

Останні 5 (рисунок 2.7):

>tail(flights[, c("dep_time", "dep_delay", "arr_time", "carrier", "tailnum")], 5)

```
# A tibble: 5 × 5
  dep_time dep_delay arr_time carrier tailnum
     <int>     <dbl>    <int> <chr>   <chr>
1       NA        NA       NA 9E      NA
2       NA        NA       NA 9E      NA
3       NA        NA       NA MQ      N535MQ
4       NA        NA       NA MQ      N511MQ
5       NA        NA       NA MQ      N839MQ
```

Рисунок 2.7 – Останні 5 рядків таблиці

Обчислимо середній час затримки прибуття (mean_arr_delay) і відправлення (mean_dep_delay) для різних перевізників (carrier):

Середня затримка прибуття (рисунок 2.8):

>mean_arr_delay<- tapply(flights$arr_delay, flights$carrier, mean, na.rm = TRUE)

>mean_arr_delay

```
        9E         AA         AS         B6         DL         EV         F9         FL         HA         MQ         OO         UA
 7.3796692  0.3642909 -9.9308886  9.4579733  1.6443409 15.7964311 21.9207048 20.1159055 -6.9152047 10.7747334 11.9310345  3.5580111
        US         VX         WN         YV
 2.1295951  1.7644644  9.6491199 15.5569853
>
```

Рисунок 2.8 – Середня затримка прибуття

Середня затримка відправлення (рисунок 2.9):

>mean_dep_delay<- tapply(flights$dep_delay, flights$carrier, mean, na.rm = TRUE)

>mean_dep_delay

```
        9E         AA         AS         B6         DL         EV         F9         FL         HA         MQ         OO         UA         US
16.725769  8.586016  5.804775 13.022522  9.264505 19.955390 20.215543 18.726075  4.900585 10.552041 12.586207 12.106073  3.782418
        VX         WN         YV
12.869421 17.711744 18.996330
```

Рисунок 2.9 - Середня затримка відправлення

5. Приклад для виконання 5 завдання, тобто згенерувати data.frame з трьома стовпцями і 100 рядками. Перетворити дані з широкого в довгий формат. Встановити пакет reshape2.

Встановимо пакет reshape2:

```
>install.packages("reshape2")
>library(reshape2)
```

Створюється дата фрейм з трьома стовпцями і 100 рядками:

```
>set.seed(123)
>data <- data.frame(
+     ID = 1:100,
+     Value1 = rnorm(100),
+     Value2 = rnorm(100),
+     Value3 = rnorm(100)
+ )
```

Отримаємо такий дата фрейм, що зображено на рисунку 2.10.

	ID	Value1	Value2	Value3
1	1	-0.560475647	-0.71040656	2.19881035
2	2	-0.230177489	0.25688371	1.31241298
3	3	1.558708314	-0.24669188	-0.26514506
4	4	0.070508391	-0.34754260	0.54319406
5	5	0.129287735	-0.95161857	-0.41433995
6	6	1.715064987	-0.04502772	-0.47624689
7	7	0.460916206	-0.78490447	-0.78860284
8	8	-1.265061235	-1.66794194	-0.59461727
9	9	-0.686852852	-0.38022652	1.65090747
10	10	-0.445661970	0.91899661	-0.05402813

Рисунок 2.10 - Створюється дата фрейм

Для перетворення у довгий формат скористаємося функцією melt:

```
>data_long<- melt(data, id.vars = "ID", variable.name = "Variable", value.name =
"Value")
```

Для відображення виведемо перші рядки перетворених даних:

```
>head(data_long)
  ID Variable       Value
```

1 1 Value1 -0.56047565

2 2 Value1 -0.23017749

3 3 Value1 1.55870831

4 4 Value1 0.07050839

5 5 Value1 0.12928774

6 6 Value1 1.71506499

Допоміжні джерела інформації:

1. https://stackoverflow.com/questions/1727772/quickly-reading-very-large-tables-asdataframes/

2. https://stackoverflow.com/questions/57047338/split-dataset-per-rows-into-smaller-files-in-r

3. https://www.tutorialspoint.com/big_data_analytics/introduction_to_sql.htm

4. https://rstudio-pubs-static.s3.amazonaws.com/378109_3e5dfcb6c47a44fbbb8b420902cd342f.html

Контрольні запитання до лабораторної роботи 2:

1. Що розуміють під поняттям масив?
2. За допомогою якої функції створюється масив в R?
3. Що розуміють під поняттям фрейм?
4. Що є основними структурними елементами фрейму даних?
5. Які основні операції над даними в фреймах?
6. Для яких цілей застосовують пакет artifact purrr?
7. Для яких цілей застосовують пакети data.table, sqldf, ff?
8. В чому полягають переваги застосування data.table?
9. Як встановити і завантажити бібліотеки sqldf і nycflights13?
10. Для яких цілей застосовують бібліотеки sqldf і nycflights13?
11. Як перетворити дані з широкого в довгий формат?
12. Для якого перетворення застосовують функцію melt?
13. Що представляє собою пакет reshape2?

Лабораторна робота №3

Тема: Підготовка вихідних даних

Мета: Застосовуючи основні характеристики мови R навчитись робити підготовку вихідних даних

Завдання:

1. Згенерувати вектор (масив, таблиця даних) і додати в нього елементи NA. Очистити дані від NA з використанням функції is.na ().
2. Згенерувати таблицю даних з числовими і текстовими стовпцями. Очистити дані функцією complete.cases ().
3. Згенерувати числову таблицю даних з пропусками. З використанням функції preProcess з пакета caret заповнити пропуски передбаченими значеннями (середнє,медіана).
4. Згенерувати два числових набори даних, додати в них викиди. З використанням функції boxplot() виявити викиди і видалити їх (джерело 1).
5. Згенеруйте таблицю даних, в якій дублюються рядки. Видаліть рядки з використанням функцій unique (), duplicated (). Порівняйте результати (джерело 2).
6. Обробити пропуски в даних з використанням пакета mice.
7. Розібрати приклад з мультиколінеарності (джерело 3).

Теоретичні відомості:

R – найпоширеніша мова програмування у світі, що використовується для статистики [1].

Корисні клавіатурні скорочення у RStudio:

Ctrl + Shift + C - закоментувати/розкоментувати виділений фрагмент коду.

Ctrl + Enter - відправляє активний рядок із текстового редактора в консоль.

Tab або Ctrl + Space - натисніть після того, як почали набирати назву функції або змінної, і з'явиться список автопідстановки. Це допомагає друкувати код швидко та з меншою кількістю помилок.

Вбудовані константи R: NA, NULL, NAN, Inf

-NA – англ “not available”. Коли об'єкт був, та його властивість не виміряли чи не записали.

-NULL - пусто - просто нічого немає.

-NaN - "not a number" Inf - "infinity" – нескінченність.

Наприклад, вектор із пропущеним значенням:

rbow_1 <- c("red", "orange", NA, "green", "blue", "violet")

При роботі з даними часто пусті та невизначені значення заважають, псують роботу при підрахунках, тому існують методи як цього уникнути. Коли виникає проблема пропущених значень, найбільш швидкий спосіб її вирішення - прості заміни: нулями, середнім, модою і т.д. Однак жоден із цих методів не гнучкий і може привести до невідповідностей даних.

Можна використовувати функцію is.na() R для перевірки пропущених значень у векторах і кадрах даних.

У наступному прикладі показано, як використовувати функцію is.na() для перевірки пропущених значень у векторі:

x <- c(3, 5, 5, **NA**, 7, **NA**, 12, 16)

#check if each individual value is NA

is. na (x)

[1] FALSE FALSE FALSE **TRUE** FALSE **TRUE** FALSE FALSE

#count total NA values

sum(is. na (x))

[1] 2

#identify positions of NA values

which(is. na (x))

[1] 4 6

Розглянемо простий випадок позбавлення від NA:

x<-c(7, 2, NA, 8, NA, 9, 1)

> bad<-is.na(x)

> x[!bad]

[1] 7 2 8 9 1

Простий варіант виконання завдання №1, наприклад:

```
>setwd('D:/BigData/')
>vector <- c(1, 2, NA, 4, NA, 6)
>cat(vector)
1 2 NA 4 NA 6
>clean_vec<- vector[!is.na(vector)]
>cat(clean_vec)
1 2 4 6
```

Є можливим використовувати функцію complete.cases() в R, щоб видалити пропущені значення у векторі, матриці або фреймі даних. Ця функція використовує наступний базовий синтаксис:

```
#видалити відсутні значення з вектора
x <- x[complete.cases(x)]
#видалити рядки з відсутніми значеннями в будь-якому стовпці кадру даних
df <- df[complete.cases(df), ]
#видалити рядки з NA в певних стовпцях кадру даних
df <- df[повний. case(df[ , c('col1', 'col2', ...)]), ]
```

Наприклад:

```
x<- c("a", "b", NA, "d", NA, "f", "r", NA, "ya")
x <- x[complete.cases(x)]
x
[1] "a"  "b"  "d"  "f"  "r"  "ya"
```

Простий варіант виконання завдання №2, наприклад (рисунки 3.1 та 3.2):

```
>data <- data.frame()
>data <- data.frame(numcolomn = c(1, 2, NA, 4, NA, 6), textcolomn = c("A", "B", NA, "D", NA, "F"))
>print(data)
```

Результат виконання на рисунку 3.1.

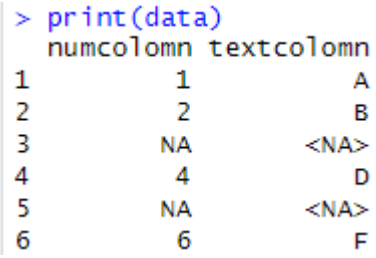


Рисунок 3.1 – Результат виконання

>cleand<- data[complete.cases(data),]

>print(cleand)

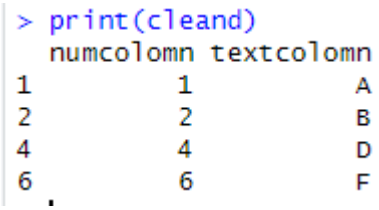


Рисунок 3.2 – Результат виконання після print(cleand)

Пакет MICE допоможе замінити пропущені значення, використовуючи різноманітні техніки, в залежності від даних, з якими працюють.

Наприклад:

Для перевірки треба створити випадковий блок даних, спеціально внести в нього декілька пропущених значень:

install.pckages(“mice”)

require(mice)

dataset2 <- mice(dataset)

dataset2<-complete(dataset2)

summary(dataset2)

На жаль, після набору даних виникають не тільки пусті комірки. Дуже часто зустрічаються просто помилки. Часто це помилки, що можуть виникнути при ручному наборі. Якщо даних небагато, то можна спробувати виявити такі помилки вручну. Гірше, якщо обсяг даних великий, наприклад, більше тисячі записів. В цьому випадку можуть допомогти методи обробки даних, перш ніж ті, що розраховані на виявлення викидів (outliers). Найпростіший із них знаходження мінімуму і максимуму, а для номінальних даних побудова таблиці частоти. На жаль, такі методи допомагають лише частково.

Пакет **caret** (от "*classification and regression training*") - класифікаційне та регресійне навчання. На даний час caret став одним із найбільш популярних інструментів серед користувачів R, що займаються розробкою передсказальних моделей.

Пакет caret володіє такими відмінними особливостями:

- використання команди універсального синтаксису, поза залежністю від вихідної функції синтаксису, реалізуючої той чи інший алгоритм;

- автоматизоване визначення оптимальних значень гіперпараметрів моделей ("параметрів налаштування"), які зазвичай неможливо вирахувати аналітично;

- можливість організації паралельних вичислень, значно прискорюючих процес навчання моделей.

До складу caret входять функції, що сприяють реалізації повного циклу розробки передбачуваних моделей.

Одна з них - preProcess()- виконує задані користувачем перетворення вихідних значень предикторов, необхідних коректного використання тих чи інших алгоритмів навчання моделей. Зокрема, є можливість виконати стандартизацію значень числових змінних, перетворення вихідних даних методом головних компонентів, а також перетворення методом "просторових знаків" ("spatial sign transformation"), яке буває особливо корисним за наявності багатовимірних "викидів".

У будь-якому виборі є дві найбільш загальні характеристики: центр (центральна тенденція) і розкид (розмах). В якості центру частіше всього використовується середнє і медіана, а в якості розкидання стандартне відхилення і квартили. Середнє відрізняється від медіани перш за все, що воно добре працює в основному тоді, коли розподіл даних близько до нормального. Медіана не залежить від характеристик розподілу, як свідчать статистики, вона більш робастна (стійка).

Крім медіани, для оцінки властивостей вибірки дуже корисні квартілі, тобто ті значення, які відсікають відповідно 0%, 25%, 50%, 75% та 100% від усього розподілу даних. Можна поняття квартиль розширити і запровадити

спеціальний термін для значення, що відсікає будь-який відсоток упорядкованого розподілу (не обов'язково по чвертях), це називається «квантиль». Квантилі використовуються, наприклад, для аналізу даних на нормальність.

Простий варіант виконання завдання №3, тобто необхідно згенерувати числову таблицю даних з пропусками. З використанням функції preprocess з пакета caret заповнити пропуски передбаченими значеннями (середнє, медіана).

Наприклад, для початку встановимо та завантажимо бібліотеку caret:

```
install.packages("caret")
library(caret)
```

Потім генерується таблиця даних (рисунок 3.3):

```
Set.seed(123)
data.frame(numcolomn = c(1, 2, NA, 4, NA, 6), textcolomn = c(A, B, NA, D, NA, F))
```

	numcolomn	textcolomn
1	1	A
2	2	B
3	NA	<NA>
4	4	D
5	NA	<NA>
6	6	F

Рисунок 3.3 – Сформована таблиця

Заповнюються пропуски середніми значеннями (рисунок 3.4):

```
>prer<- function(data) {data$numcolomn[is.na(data$numcolomn)] <- mean(data$numcolomn, na.rm = TRUE)
+ return(data)
+ }
>data_mean_filled<- prer(data)
>print(data_mean_filled)
```

	numcolomn	textcolomn
1	1.00	A
2	2.00	B
3	3.25	<NA>
4	4.00	D
5	3.25	<NA>
6	6.00	F

Рисунок 3.4– Сформована таблиця з середніми значеннями

Заповнимо пропуски медіаною (рисунок 3.5):

```
>prer_mediana<- function(data) {data$numcolomn[is.na(data$numcolomn)] <-
median(data$numcolomn, na.rm = TRUE)
+ return(data)
+ }
>data_median_filled<- prer_mediana(data)
>print(data_median_filled)
```

	numcolomn	textcolomn
1	1	A
2	2	B
3	3	<NA>
4	4	D
5	3	<NA>
6	6	F

Рисунок 3.5– Сформована таблиця з медіаною

Діаграми розмахів, або "ящики з вусами" (англ. box-whisker plots), отримали свою назву за характерний вигляд: точку або лінію, що відповідає медіані або середній арифметичній, оточує прямокутник ("ящик"), довжина якого відповідає одному з показників розкиду чи точності оцінки генерального параметра. Додатково від цього прямокутника відходять "вуса", що також відповідають за довжиною одному з показників розкиду або точності. Графіки цього дуже популярні, оскільки дозволяють дати дуже повну статистичну характеристику аналізованої сукупності. Крім того, діаграми розмаху можна використовувати для візуальної експрес-оцінки різниці між двома та більше групами (наприклад, між датами відбору проб, експериментальними групами, ділянками простору тощо). В R для побудови діаграм розмахів служить функція boxplot() [8].

Функція boxplot() служить для побудови діаграм розмахів в R.

Один з прикладів виконання завдання №4, тобто згенерувати два числових набори даних, додати в них викиди. З використанням функції boxplot() виявити викиди і видалити їх:

Генеруються два набори даних:

```
>set.seed(123)
>data1 <- rnorm(100)
```

>data2 <- rnorm(100)

Додаємо викиди:

>data1 <- c(data1, rnorm(10, mean=5))

>data2 <- c(data1, rnorm(10, mean=5))

Виявлення викидів:

```
>boxplot1 <- boxplot(data1,plot=FALSE)
>boxplot2 <- boxplot(data1,plot=FALSE)
```

Plot має значення false інакше виникне помилка:

Error in plot.new() : figure margins too large

Але щоб їх переглянути можемо записати їх в pdf.

```
>pdf("myplot.pdf", width = 8, height = 6)
>boxplot1 <- boxplot(data1)
>boxplot2 <- boxplot(data2)
>dev.off()
```

Тепер видаляються викиди:

```
>data1_clean <- data1[-which(data1 %in% boxplot1$out)]
>data2_clean <- data2[-which(data2 %in% boxplot2$out)]
```

Та перевіряються результати знову з пдф файлом.

```
>pdf("myplot.pdf", width = 8, height = 6)
>boxplot(data1_clean, data2_clean, names=c("Data 1", "Data 2"))
>dev.off()
```

Функції unique () застосовують щоб отримати унікальні значення у стовпці кадру даних.

Функція duplicated () - визначає, які елементи вектора або кадру даних є дублікатами елементів з меншими індексами, і повертає логічний вектор, що вказує, які елементи (рядки) є дублікатами.

Предиктори - це незалежні змінні.

Під мультиколінеарністю розуміють наявність лінійної залежності між двома або більше факторними (незалежними) змінними у регресійній моделі [9].

Якщо два предиктори у моделі є однією змінною але у різних метричних шкалах. Наприклад, ріст людини у сантиметрах і ріст людини у дюймах. Коефіцієнт кореляції між двома змінними буде дорівнювати 1. Аби уникнути мультиколінеарності, у модель має вступувати лише одна із двох змінних.

Серед ознак мультиколінеарності [9]: велике значення коефіцієнту детермінації поряд з незначущістю коефіцієнтів моделі; велике значення парних коефіцієнтів кореляції незалежних (факторних) змінних.

Методи виявлення мультиколінеарності: Алгоритм Фаррара-Глобера, VIF

Допоміжні джерела інформації:

1. https://www.r-bloggers.com/outlier-detection-and-treatment-with-r/
2. https://stackoverflow.com/questions/13967063/remove-duplicated-rows
3. https://datascienceplus.com/multicollinearity-in-r/

Контрольні запитання до лабораторної роботи 3:

1. Які є вбудовані константи в R?
2. Що представляє собою NA?
3. Чому уникають пустих та невизначених значень в таблицях?
4. Яку функцію застосовують для перевірки пропущених значень у векторах і кадрах даних в R?
5. Що виконується у наступних рядках, якщо спочатку: bad<-is.na(x) , а потім: x[!bad]?
6. За допомогою якої функції можна видалити пропущені значення у векторі, матриці або фреймі даних?
7. Що представляє собою пакет caret?
8. Що виконує функція preProcess()?
9. Що виконує функція boxplot()?
10. Що виконує функція unique ()?
11. Що виконує функція duplicated ()?
12. Що представляє собою пакет mice в R?
13. Що розуміють під терміном мультиколінеарність?

14. Які найбільш загальні характеристики є у будь-якому виборі?

Лабораторна робота №4

Тема: Обробка даних. Вибір ознак (FeatureSelection)

Мета: Навчитись виконувати обробку даних, застосовувати вибір ознак при роботі з даними в R

Завдання:

1. Встановити пакет CARET, виконати команду names (getModelInfo ()), ознайомитися зі списком доступних методів вибору ознак. Виконайте графічний розвідувальний аналіз даних з використання функції featurePlot () для набору даних з довідкового файлу пакета CARET:

x <- matrix (rnorm (50 * 5), ncol = 5)

y <- factor (rep (c ("A", "B"), 25))

Зберегти отримані графіки в * .jpg файли. Зробити висновки.

2. З використанням функцій з пакета Fselector [11] визначити важливість ознак для рішення задачі класифікації. Використовувати набір data(iris). Зробити висновки.

3. Встановіть пакет Boruta і проведіть вибір ознак для набору даних data("Ozone") [12, 13]. Побудувати графік boxplot, зробити висновки.

Теоретичні відомості:

Пакет CARET - Classification And REgression Training - розроблений для комбінування моделей навчання та прогнозування. У пакеті є кілька алгоритмів, придатних для різних завдань.

Пакет CARET дозволяє підібрати оптимальні параметри для алгоритму за допомогою контрольованих експериментів. Метод перехресного пошуку, реалізований в цьому пакеті, виконує пошук параметрів, комбінуючи різні методи оцінки продуктивності моделей. Після перебору всіх можливих комбінацій метод перехресного пошуку знаходить комбінацію, яка дає найкращі результати.

Можливо використовувати пакет caret для полегшення кодування [10], але концепція застосовується до будь-якої моделі на будь-якій мові статистичного програмування. Caret дозволяє легко перемикати моделі в сценарії, не

змінюючи багато коду. Ви можете легко написати цикл і виконати його через майже 170 моделей, які зараз підтримує пакет, змінивши лише одну змінну.

Щоб отримати повний список моделей, які підтримуються caret, скористайтеся функцією getModelInfo:

```
library(caret)
names(getModelInfo())
```

Як буде видно з результату, доступних моделей достатньо, щоб задовольнити більшість потреб. Деякі підтримують або подвійне використання, тоді як інші або лише класифікацію, або регресію. Ви можете перевірити тип, який підтримує модель, використовуючи ту саму функцію getModelInfo:

```
getModelInfo()$glm$type
# "Регресія" "Класифікація"
```

Тут видно, що glm підтримує як регресію, так і класифікацію.

Якщо розглядать етапи статистичного моделювання, то першим етапом є розвідувальний аналіз структури даних. Термін «структура даних» об'єднує характер розподілу сукупності за ознаками, що описують об'єкт моделювання, і характер взаємозв'язків між цими ознаками.

Термін «розвідувальний аналіз» (*exploratory data analysis - EDA*) був введений Дж. Тьюкі, він же сформулював основні його завдання:

- максимальне проникнення в дані;
- вибір найважливіших ознак:
- аналіз основних структур;
- виявлення відхилень і аномалій;
- перевірка основних гіпотез щодо законів розподілу і взаємозв'язків;
- апробація моделей.

На етапі розвідувального аналізу формується уявлення про тип даних, оцінюється їх однорідність, з'ясовується структура об'єкта моделювання, виявляються взаємозв'язки між ознаками [7].

Графічний аналіз характеру розподілу даних та їх взаємної залежності, наявність викидів та інших аномальних ситуацій може бути виконано за допомогою функції featurePlot().

FSelector – це пакет, що містить функції для вибору атрибутів із заданого набору даних і атрибута призначення.

Цей пакет містить:

•Алгоритми для фільтрації атрибутів: cfs, chi.squared, information.gain, gain.ratio, symmetri-cal.uncertainty, linear.correlation, rank.correlation, oneR, relief, consistency, random.forest.importance.

•Алгоритми для обгортання класифікаторів і простору підмножини атрибутів пошуку: best.first.search, back-ward.search, forward.search, hill.climbing.search.

•Алгоритм для вибору підмножини атрибутів на основі ваги атрибутів: cutoff. k, cut-off.k.percent, cutoff.biggest.diff.

•Алгоритм створення формул: as.simple.formula.

Класифікація є найбільш простою і водночас найбільш часто розв'язуваною задачею DataMining.

Класифікація–системний розподіл досліджуваних предметів, явищ, процесів за родами, видами, типами, з якими-небудь істотними ознаками для зручності їх дослідження; угрупування вихідних понять і розташування їх у певному порядку, що відбиває ступінь цієї схожості.

Класифікація–упорядкована за деяким принципом множина об'єктів, які мають подібні класифікаційні ознаки (одна або декілька властивостей), обраних для визначення схожості або відмінності між цими об'єктами.

Процес класифікації складається з двох етапів: конструювання моделі та її використання.

Оскільки пошук хороших наборів даних та їх розмітка — важке завдання, на допомогу приходять вже зібрані та розмічені набори даних, для яких часто вже опубліковані результати якихось алгоритмів, і можна оцінити, наскільки добре працює модель, що досліджується.

Набір iris - невеликий набір даних для задачі класифікації. У цьому наборі даних представлено по 50 описів квіток одного з трьох типів - Ірис щетинистий (Iris setosa), Ірис віргінський (Iris virginica) та Ірис різнокольоровий (Iris versicolor). Для кожної квітки виміряно чотири величини — довжина

чашолистки (англ. sepal length), ширина чашолистки (sepal width), довжина пелюстки (англ. petal length), ширина пелюстки (англ. petal width). Всі квітки промарковані одним із трьох типів, що дозволяє тестувати на ньому алгоритми класифікації.

Оскільки набір даних iris є вбудованим набором даних R, ми можемо завантажити його за допомогою наступної команди: data(iris)

Пакет Boruta - це алгоритм обгортки для вибору всіх відповідних функцій. Він знаходить релевантні функції, порівнюючи важливість оригінальних атрибутів із випадковою важливістю, оціненою за допомогою їх змінених копій (тіней).

Набір даних Ozone можна отримать після встановлення пакета mlbench.

Приклад виконання завдання №1:

```
>library(caret)
>print(names(getModelInfo()))
```

Виведеться список доступних методів вибору даних кілька з них представлено на рисунку 4.1:

```
  [1] "ada"               "AdaBag"            "AdaBoost.M1"       "adaboost"
  [5] "amdai"             "ANFIS"             "avNNet"            "awnb"
  [9] "awtan"             "bag"               "bagEarth"          "bagEarthGCV"
 [13] "bagFDA"            "bagFDAGCV"         "bam"               "bartMachine"
 [17] "bayesglm"          "binda"             "blackboost"        "blasso"
 [21] "blassoAveraged"    "bridge"            "brnn"              "BstLm"
 [25] "bstSm"             "bstTree"           "C5.0"              "C5.0Cost"
 [29] "C5.0Rules"         "C5.0Tree"          "cforest"           "chaid"
 [33] "CSimca"            "ctree"             "ctree2"            "cubist"
 [37] "dda"               "deepboost"         "DENFIS"            "dnn"
 [41] "dwdLinear"         "dwdPoly"           "dwdRadial"         "earth"
 [45] "elm"               "enet"              "evtree"            "extraTrees"
 [49] "fda"               "FH.GBML"           "FIR.DM"            "foba"
 [53] "FRBCS.CHI"         "FRBCS.W"           "FS.HGD"            "gam"
```

Рисунок 4.1 - Список доступних методів

Далі створюється набір даних:

```
>x<- matrix(rnorm(50 * 5), ncol = 5)
>y <- factor(rep(c("A", "B"), 25))
>data <- data.frame(x, y)
```

Далі виконаємо розвідувальний аналіз і збережемо його в форматі jpg:

```
>jpeg("featplot.jpg")
>featurePlot(
+     x = data[, -ncol(data)],
```

```
+     y = data$y,
+     plot = "pairs",
+     strip = strip.custom(strip.names = TRUE),
+     col = c("blue", "red"),
+     main = "Графічнийрозвідувальнийаналізданих"
+ )
>dev.off()
```

В результаті отримаємо рисунок 4.2:

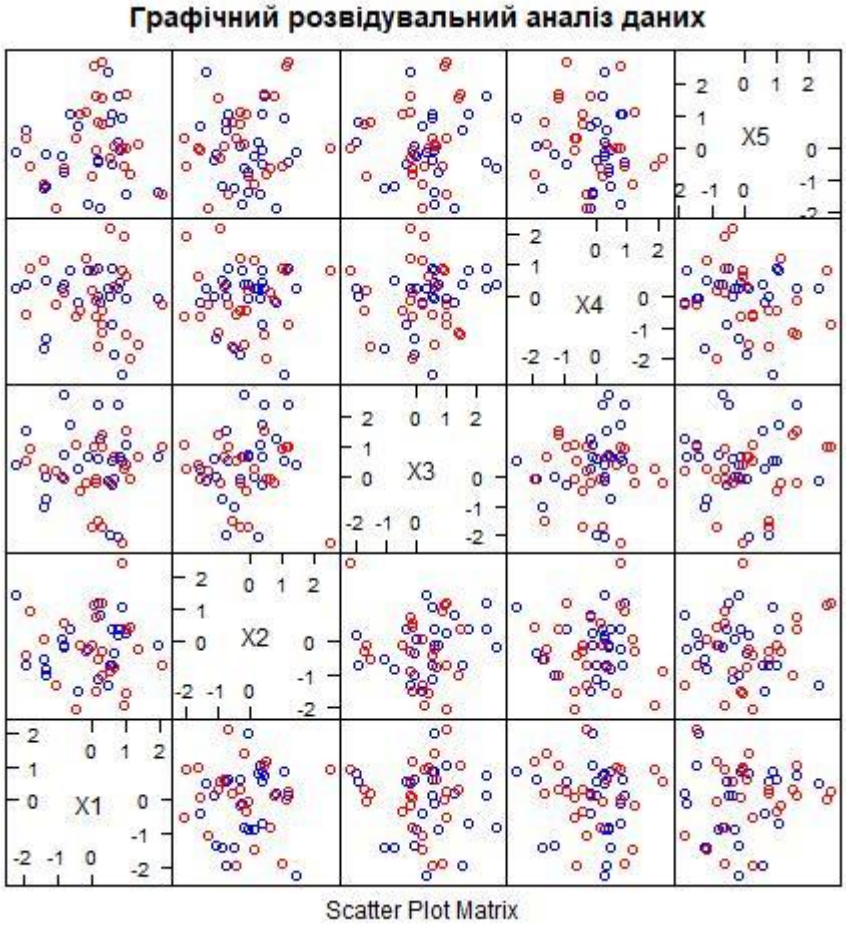


Рисунок 4.2 – Графічний розвідувальний аналіз даних

Змінні X1, X2, X3, X4 та X5 мають розподіл, який схожий на нормальний, оскільки точки на діаграмах розсіювання формують вигляд еліпсу, що є характерною особливістю нормального розподілу. Для деяких пар змінних важко визначити в одному просторовому розмірі, для класифікації цих даних може бути необхідним використання багатовимірних методів.

Допоміжні джерела інформації:

1. https://topepo.github.io/caret/train-models-by-tag.html#implicit-feature-selection
2. https://miningthedetails.com/blog/r/fselector/
3. https://www.jstatsoft.org/article/view/v036i11/v36i11.pdf
4. https://www.datacamp.com/community/tutorials/feature-selection-R-boruta
5. http://ai.stanford.edu/~ronnyk/wrappersPrint.pdf
6. https://posibniki.com.ua/post-rozviduvalnii-analiz-strukturi-danih-instrumenti-i-strategiyi

Контрольні запитання до лабораторної роботи 4:

1. Що можна виконати за допомогою команди names (getModelInfo ())?
2. Для яких цілей застосовують функцію featurePlot ()?
3. Що представляє собою пакет Fselector?
4. Що представляє собою набір data (iris)?
5. Що представляє собою пакет Boruta?
6. Що розуміють під поняттям класифікації?
7. З яких етапів складається процес класифікації?

Лабораторна робота №5

Тема: Обробка даних. Вибір екземплярів (InstanceSelection)

Мета: Навчитись виконувати обробку даних, застосовувати вибір екземплярів при роботі з даними в R

Завдання:

1. Виконайте класифікацію k-найближчих сусідів з використанням функції knn () з пакету class на наборі даних iris [16]. Проведіть нормалізацію даних, розділіть вибірку на навчальну і тестову. Оцініть побудовану модель з використанням функції CrossTable () з пакету gmodels. Побудуйте матрицю помилок і діагональну оцінку якості прогнозу (diagonalmarkqualityprediction).

2. Розгляньте приклад реалізації методу опорних векторів з використанням функції svm () з пакету e1071. Побудуйте лінійний класифікатор для прогнозування. Для підбору параметрів моделі виконайте перехресну перевірку з розподілом вихідної вибірки на 10 рівних частин (cross = 10).

3. Виконайте розрахунок головних компонент з використанням пакета vegan () і його функції rda (). Побудуйте ординаційну діаграму методом PCA і зробіть висновки.

Теоретичні відомості:

З причини того, що класифікація є найбільш простою і водночас найбільш часто розв’язуваною задачею DataMining, як розглядалось в попередній роботі, то в цій роботі продовжимо працювати з нею.

Основні процедури класифікації можна поділити на дві групи: методи класифікації "з учителем" та методи класифікації "без учителя" [3].

В основі методів класифікації "з учителем" лежить задача віднесення деякого об’єкта до одного з уже відомих класів. При цьому вхідною інформацією мають бути дані про різні характеристики, що пов’язані з приналежністю до відповідного класу (навчальний набір даних). Методи даної групи включають методи дискримінантного аналізу (лінійного – функція lda() та квадратичного – функція qda() пакету MASS) [3], дерева рішень (функції

пакету rpart), випадковий ліс (функції пакету randomForest), та нейронні мережі (функції пакету nnet).

Методи класифікації "без учителя" ще називають методами кластерного аналізу. Відповідні процедури дозволяють розбити сукупність об’єктів на окремі класи, приналежність до яких невідома до проведення класифікації. Методи даної групи включають дендограми (функція hclust()) та метод к-найближчих сусідів (функція knn() пакету class).

Іриси Фішера – найпопулярніший у статистичній літературі набір даних, що часто використовується для ілюстрації роботи різних алгоритмів класифікації. Він дозволяє побудувати добрий класифікатор при мінімумі вихідних ознак. Маємо по 50 описів квіток одного з трьох типів - Iris setosa, Iris virginica та Iris versicolor. Для кожної квітки маємо чотири величини, (що розглядалось вже в попередній роботі) - sepal length, sepal width, petal length, petal width.

Застосовується найпростіший, але досить ефективний алгоритм найближчих сусідів (k nearest neighbors), або kNN-регресію. В основі методу kNN-класифікатора лежить гіпотеза компактності, яка передбачає, що об'єкт d, що тестується, матиме таку ж мітку класу, як і навчальні об'єкти в локальній області його найближчого оточення. У варіанті 1NN аналізований об'єкт відносять до певного класу в залежності від інформації про його найближчого сусіда. У варіанті kNN кожен об'єкт відноситься до переважного класу найближчих сусідів, де k - параметр алгоритму. Вирішальні правила в методі kNN визначаються межами суміжних сегментів діаграми Вороного (Voronoi resselation), що розділяє площину на n опуклих багатокутників, кожен з яких містить один і лише один об'єкт навчальної вибірки [16]. У p-вимірних просторах межі рішень складаються вже із сегментів (p-1)-вимірних напівплощин, утворених опуклими багатогранниками Вороного.

Для класифікації у середовищі R за допомогою найближчих сусідів може бути використана функція knn() з базового пакета class і kknn() з одноіменного пакета з розширеними можливостями налаштувань близькості та форми міри ядерної функції.

Щоб створити модель необхідно завантажити пакет class і викликати функцію knn(). Наприклад:

```
#install.packages("class")
library("class")
iris_mdl = knn(train=iris_train, test=iris_test, cl=iris_train_labels, k=3)
```

Функція knn приймає як аргументи навчальний набір даних і тестовий набір даних.

Щоб оцінити модель, використовуйте функцію CrossTable() у пакеті gmodels. #install.packages("gmodels")

```
library("gmodels")
CrossTable(x=iris_test_labels, y=iris_mdl, prop.chisq = FALSE)
```

Перед переходом до самих метрик необхідно ввести важливу концепцію для опису цієї метрики в термінах помилок класифікації — матриця помилок (confusion matrix)

Наприклад:

Допустимо, що у нас є два класи і алгоритм, що передбачає до якого з належних класів відноситься кожен об'єкт, тоді матриця помилок класифікації буде виглядати як представлено В таблиці 5.1.

Таблиця 5.1 – Матриця помилок класифікації

	y=1	y=0
Y0=1	True Positive (TP)	False Positive (FP)
Y0=0	False Negative (FN)	True Negative (TN)

Y0 – в цьому випадку - відповідь алгоритму на об’єкті, а y - це справжня мітка класу на цьому об'єкті. Отже, помилки класифікації розглядаються двох типівв: False Negative (FN) и False Positive (FP).

Метод опорних векторів - це метод аналізу даних для класифікації та регресійного аналізу за допомогою моделей з керованим навчанням з пов'язаними алгоритмами навчання, які називаються опорно-векторними машинами (ОВМ, англ. support vector machines, SVM, також опорно-векторними мережами, англ. support vector networks [6])

Основна ідея класифікатора на опорних векторах полягає в тому, щоб побудувати розділяючу поверхню за допомогою лише невеликї підмножини точок, розташованих у зоні, критичної для поділу, тоді як інші вірно класифіковані спостереження, що навчаються, цієї зони ігноруються (точніше, є «резерватором» для оптимізації алгоритма).

Для реалізації методу опорних векторів в середовищі R зазвичай використовується функція svm() з пакета e1071.

Розглянемо побудоване за допомогою цієї функції лінійного класифікатора для прогнозування способу виробництва скла за його хімічним складом. Для підбору параметрів моделі задаємо перехрестну перевірку з розділом вихідного вибору на 10 рівних частин (хрестик = 10):

```
DGlass <- read.table(file = "Glass.txt", sep = ",",
                     header = TRUE, row.names = 1)
DGlass$FAC <- as.factor(ifelse(DGlass$Class == 2, "C2", "C1"))
svm.all <-  svm(formula = FAC ~ ., data = DGlass[, -10],
                cross = 10, kernel = "linear")
(table(Факт = DGlass$FAC, Прогноз = predict(svm.all)))
…
Acc = mean(predict(svm.all) == DGlass$FAC)
paste("Точність=", round(100*Acc, 2), "%", sep = "")
## [1] "Точность=74.23%"
```

Ординація коротко - є знаходження таких координатних осей на площині, відносно яких можливе оптимальне процілювання багатомірних аналізованих об'єктів. Ординація пов’язана з пошуком найбільш “природних” системних закономірностей і особливо ефективна при наявності комплексу взаємно скорельованих спливаючих факторів. До цього часу в середовищі R розроблено та реалізовано значну кількість алгоритмів згортання інформаційного простору, зокрема: метод головних компонентів PCA, оперуючий з коваріаційною (кореляційною) матрицею; метод головних координат PCoA і неметричне багатомірне шкалування (NMDS, nonmetric multidimensional scaling), що виконують послідовну процедуру перетворення будь-якої матриці відстаней;

аналіз відповідності або корреспондентний аналіз (CA,respondence analysis), заснований на ітераційній процедурі зустрічного усереднення взвешуваних коефіцієнтів для об'єктів і змінних ; функції багатомірного факторного аналізу, включаючи версії для ієрархічних і змішаних даних, представлені в пакеті FactoMineR.

Класичний метод зниження розмірності аналізу даних є основним компонентом (PCA, Principal Components Analysis), який широко використовується в різних галузях науки і техніки.

Зниження розмірності вихідного простору методом PCA можна представити як послідовний, ітеративний процес, який можна обернути на будь-якому кроці u. У результаті ортогональності системи координат головних компонентів (тобто фактично, їх взаємної незалежності) немає необхідності перебудовувати матриці рахунків (scores) Fi навантажень (loadings) U при зміні числа компонентів: наступні стовпці або рядки просто додаються або віднімаються.

Використовуємо як приклад ідентифікацію типу шибок (Glass Identification) за вибіркою, представленою в колекції баз даних.

У ході криміналістичної експертизи найдрібніших осколків скла можна визначити їх оптичну рефракцію (RI) і зробити хімічний аналіз вмісту оксидів основних елементів. Постає питання, чи можна за цими даними виконати прогноз типу виробництва скла Class. Вихідні дані представлені у файлі Glass.txt

У середовищі R розрахунок головних компонент може бути здійснений з використанням функцій princomp() або prcomp(), але ми орієнтуватимемося на різноманітність можливостей пакета vegan() та його функцію rda(). Протокол аналізу обмежимо чотирма головними компонентами, що пояснюють 98% загальної варіації даних:

```
library(vegan)
Y <- as.data.frame(DGlass[, 1:9])
mod.pca <- vegan::rda(Y ~ 1)
head(summary(mod.pca))
```

Розглядаються взаємне розташування згущень точок, що належать склам двох типів: Class = {1, 3} – віконне та автомобільне флеш-скло, і Class = 2 – тягнуте скло. Виконаємо побудову ординаційної діаграми з використанням пакету ggplot2.

```
FAC <- as.factor(ifelse(DGlass$Class == 2, 2, 1))
pca.scores <- as.data.frame(summary(mod.pca)$sites[, 1:2])
pca.scores <- cbind(pca.scores, FAC)
# Складаємо таблицю для "каркасу" точок на графику:
l <- lapply(unique(pca.scores$FAC), function(c)
      { f <- subset(pca.scores, FAC == c); f[chull(f), ]})
hull <- do.call(rbind, l)
# Включаємо в назви вісі долі поясненої дисперсії:
axX <- paste("PC1 (", as.integer(100*mod.pca$CA$eig[1]/sum(mod.pca$CA$eig)),
"%)")
axY <- paste("PC2 (", as.integer(100*mod.pca$CA$eig[2]/sum(mod.pca$CA$eig)),
"%)")
# Виведення ординаційної діаграми:
ggplot() +
  geom_polygon(data = hull,
           aes(x = PC1, y = PC2, fill = FAC), alpha = 0.4, linetype = 0) +
  geom_point(data = pca.scores,
          aes(x = PC1, y = PC2, shape = FAC, colour = FAC), size = 3) +
  scale_colour_manual(values = c('purple', 'blue')) +
  xlab(axX) + ylab(axY) + coord_equal() + theme_bw()
```

Допоміжні джерела інформації:

1. https://en.proft.me/2017/01/22/classification-using-k-nearest-neighbors-r/

2. Olvera-López, José& Carrasco-Ochoa, Jesús& Martínez-Trinidad, JoséFrancisco&Kittler, Josef. (2010). A reviewofinstanceselectionmethods. Artif. Intell. Rev. 34. 133-143.10.1007/s10462-010-9165-y.

https://mafiadoc.com/a-review-of-instance-selection-methods-softcomputing-and-_5b054f698ead0ed4758b4586.html

3. Top 10 algorithmsindatamining http://www.cs.umd.edu/~samir/498/10Algorithms-08.pdf

Контрольні запитання до лабораторної роботи 5:

1. Що розуміють під поняттям класифікація ?

2. Які переваги застосування набору даних Іриси Фішера?

3. Що лежить в основі методу kNN-класифікатора?

4. Яка функція може бути використана для класифікації у середовищі R за допомогою алгоритму найближчих сусідів?

5. Що функція knn приймає як аргументи (iris_mdl = knn(train=iris_train, test=iris_test, cl=iris_train_labels, k=3))?

6. Що представляє собою метод PCA?

7. Що представляє собою метод опорних векторів?

8. Яка функція зазвичай використовується для реалізації методу опорних векторів в середовищі R?

9. Що розуміють під ординацією?

Лабораторна робота №6

Тема: Обробка даних. Дискретизація для класифікації (Discretization)

Мета: Навчитись виконувати обробку даних, застосовувати дискретизацію для класифікації (Discretization) при роботі з даними в R

Завдання:

1. З використанням функції discretize () з пакету arules виконайте перетворення безперервної змінної в категоріальну [19] різними методами: «interval» (рівна ширина інтервалу), «frequency» (рівна частота), «cluster» (кластеризація) і «fixed»(Категорії задають межі інтервалів). Використовуйте набір даних iris. Зробіть висновки.

2. З використанням пакета discretization виконайте дискретизацию з використанням алгоритмів Chi2 і CAIM [19][21]. Використовуйте набір даних iris. Порівняйте результати і зробіть висновки.

Теоретичні відомості:

У прикладній математиці дискретизація — це процес перетворення неперервних функцій, моделей, змінних і рівнянь у дискретні аналоги. Зазвичай цей процес виконується як перший крок до того, щоб зробити їх придатними для чисельної оцінки та впровадження на цифрових комп’ютерах.

Якщо змінна може приймати будь-які значення, така змінна називається *безперервною.*

Якщо змінна може приймати тільки певні значення, така змінна називається *дискретною*.

Номінальні змінні використовують лише для якісної кваліфікації. Це значить, такі змінні можуть бути виміряні лише у термінах належності до певних класів. При цьому неможливо визначити кількість або упорядити ці класи. Типові приклади номінальних змінних – стать, національність, партія, фракція. Інколи такі змінні називають категоріальними.

Для всановлення пакета arules застосовуємо:

```
install.packages("arules")
```

Discretize обчислює перерви між інтервалами за допомогою різних методів, а потім використовує base::cut() для перетворення числових значень в інтервали, представлені як коефіцієнт.

Дискретизація може бути невдалою з кількох причин. Деякі причини є:

-Змінна містить лише одне значення. У цьому випадку змінна повинна бути відкинута або безпосередньо перетворена на фактор з одним рівнем.

-Деякі розраховані розриви не є унікальними. Це може статися для частоти методу з дуже спотвореними даними (наприклад, велика частина значень дорівнює 0). У цьому випадку неунікальні розриви видаляються з попередженням. Було б, напевно, краще подивитися на гістограму даних і визначитися з перервами для фіксованого методу.

Ця функція реалізує кілька основних методів для перетворення безперервної змінної в категоріальну змінну (фактор) за допомогою різних стратегій групування. Для зручності весь data.frame може бути дискретизованим (тобто всі числові стовпці дискретизовані) [20], Наприклад:

```
discretize(
  x,
  method = "frequency",
  breaks = 3,
  labels = NULL,
  include.lowest = TRUE,
  right = FALSE,
  dig.lab = 3,
  ordered_result = FALSE,
  infinity = FALSE,
  onlycuts = FALSE,
  categories = NULL,
  ...
)

discretizeDF(df, methods = NULL, default = NULL)
```

X	числовий вектор (неперервна змінна).
Method	метод дискретизації. Доступні: «інтервал» (однакова ширина інтервалу), «частота» (однакова частота), «кластер»

	(k-означає кластеризацію) і «фіксований» (категорії вказують межі інтервалу). Зауважте, що однакова частота не досягає ідеальних груп однакового розміру, якщо дані містять повторювані значення.
breaks, categories	або кількість категорій, або вектор із межами для дискретизації (усі значення поза межами буде встановлено як NA). категорії застаріло, замість цього використовуйте розриви.
Labels	символьний вектор; мітки для рівнів отриманої категорії. За замовчуванням мітки створюються з використанням позначення інтервалів "[a,b]". Якщо labels = FALSE, замість множника повертаються прості цілі коди.
include.lowest	логічний; чи повинен перший інтервал замикатися зліва?
Right	логічний; інтервали повинні бути закриті праворуч (і відкриті ліворуч) чи навпаки?
dig.lab	ціле число; кількість цифр, які використовуються для створення міток.
ordered_result	логічний; повернути впорядкований фактор?
Infinity	логічний; чи потрібно першу/останню межу розриву змінити на +/-Inf?
Onlycuts	логічний; повертати лише обчислені межі інтервалів?
...	для методу "cluster" додаткові аргументи передаються kmeans.
Df	data.frame; кожен числовий стовпець у data.frame є дискретизованим.
methods	іменований список списків або data.frame; іменований список містить списки параметрів дискретизації. Якщо для стовпця не вказано дискретизацію, то використовуються параметри за замовчуванням для discretize(). Примітка: імена повинні точно збігатися. Якщо вказано data.frame, то розриви дискретизації в

цьому data.frame застосовуються до df.

Default іменний список; параметри для discretize() використовуються для всіх стовпців, не вказаних у методах.

Discretize() повертає коефіцієнт, що представляє категоризовану безперервну змінну з атрибутом "discretized:breaks", що вказує на використані розриви, або "discretized:method", що вказує на використаний метод. Якщо використовується onlycuts = TRUE, повертається вектор із обчисленими межами інтервалу.

DiscretizeDF() повертає discretized data.frame.

Інші приклади:

```
data(iris)
x <- iris[,1]
### look at the distribution before discretizing
hist(x, breaks = 20, main = "Data")
def.par <- par(no.readonly = TRUE) # save default
layout(mat = rbind(1:2,3:4))
### convert continuous variables into categories (there are 3 types of flowers)
### the default method is equal frequency
table(discretize(x, breaks = 3))
hist(x, breaks = 20, main = "Equal Frequency")
abline(v = discretize(x, breaks = 3,
  onlycuts = TRUE), col = "red")
# Note: the frequencies are not exactly equal because of ties in the data
### equal interval width
table(discretize(x, method = "interval", breaks = 3))
hist(x, breaks = 20, main = "Equal Interval length")
abline(v = discretize(x, method = "interval", breaks = 3,
  onlycuts = TRUE), col = "red")
### k-means clustering
table(discretize(x, method = "cluster", breaks = 3))
```

```
hist(x, breaks = 20, main = "K-Means")
abline(v = discretize(x, method = "cluster", breaks = 3,
  onlycuts = TRUE), col = "red")
### user-specified (with labels)
table(discretize(x, method = "fixed", breaks = c(-Inf, 6, Inf),
    labels = c("small", "large")))
hist(x, breaks = 20, main = "Fixed")
abline(v = discretize(x, method = "fixed", breaks = c(-Inf, 6, Inf),
    onlycuts = TRUE), col = "red")
par(def.par)  # reset to default
### prepare the iris data set for association rule mining
### use default discretization
irisDisc <- discretizeDF(iris)
head(irisDisc)
### discretize all numeric columns differently
irisDisc <- discretizeDF(iris, default = list(method = "interval", breaks = 2,
  labels = c("small", "large")))
head(irisDisc)
### specify discretization for the petal columns and don't discretize the others
irisDisc <- discretizeDF(iris, methods = list(
  Petal.Length = list(method = "frequency", breaks = 3,
    labels = c("short", "medium", "long")),
  Petal.Width = list(method = "frequency", breaks = 2,
    labels = c("narrow", "wide"))
  ),
  default = list(method = "none")
  )
head(irisDisc)
### discretize new data using the same discretization scheme as the
###   data.frame supplied in methods. Note: NAs may occure if a new
###   value falls outside the range of values observed in the
```

```
###   originally discretized table (use argument infinity = TRUE in
###   discretize to prevent this case.)
discretizeDF(iris[sample(1:nrow(iris), 5),], methods = irisDisc)
```

Існують параметричні тести пропорції, які припускають нормальність цільових розподілів вибірки (інакше їх називають Z-тестами).

Z-тестування стосується даних, які є двійковими за своєю природою. Щоб статистично перевірити твердження щодо більш загальних категоріальних змінних, які мають більше двох різних рівнів, застосовується chi-squared test.Іноді позначається скорочено як тест $\chi 2$. Існує два поширених варіанти тесту хі-квадрат. Перший - тест хі-квадрат розподілу, також званий тестом відповідності (GOF) - використовується під час оцінки частот на рівнях однієї категоріальної змінної. Другий - тест хі-квадрат незалежності - використовується, коли досліджується зв'язок між частотами на рівнях двох таких змінних [19].

R забезпечує функцію швидкого використання для виконання тест хі-квадрат GOF. Функція chisq.test приймає вектор спостережувані частоти як його перший аргумент x. Наприклад:

```
R> chisq.test(x=hairy.tab)
```

Алгоритм CAIM (class-attribute interdependence maximization) - призначений для дискретизації безперервних даних [22].

Caim - допоміжна функція для алгоритму дискретизації caim. Наприклад:

```
a=c(3,0,3,0,6,0,0,3,0)
m=matrix(a,ncol=3,byrow=TRUE)
caim(m)
```

Допоміжні джерела інформації:

1. https://cran.r-project.org/
2. https://rdrr.io/cran/arules/src/R/discretize.R
3. https://www.rdocumentation.org/packages/discretization/versions/1.0-1.1/topics/caim

4. Kurgan, L. and Cios, K.J., 2004. CAIM Discretization Algorithm. IEEE Transactions on Knowledge and Data Engineering, 16(2):145-153

Контрольні запитання до лабораторної роботи 6:

1. Що розуміють під поняттям дискретизація?
2. Які змінні називають безперервними?
3. Які змінні називають категоріальними?
4. Що обчислюється за допомогою функції Discretize?
5. Яке значення повертає функція discretize()?
6. В яких випадках застосовується chi-squared test?
7. Для яких цілей призначений алгоритм CAIM?

Лабораторна робота №7

Тема: Організація розподілених обчислень

Мета: Навчитись розподіляти обчислення при роботі з даними в R

Завдання:

1. Встановіть пакет sparklyr, встановіть JavaVirtualMachine (JVM) [23]. Підключіться до локального Spark-кластеру. Завантажте таблицю flights з пакета nycflights13 в Spark-кластер. Виконайте запити (завдання 3, Лабораторна робота 2). Порівняйте результати,зробіть висновки.

2. Налаштуйте для використання Hadoop [24-26], підрахуйте кількість слів у файлі* .Txt с використанням HDFS. Файл згенерувати самостійно.

3. Встановіть MongoDB [27, 28]. Підключіть бібліотеку mongolite. виконайте приклад для набору iris с використання функції mongo() з прикладу з відеороліку. Збережіть код та зробіть висновки.

Теоретичні відомості:

Існують спеціалізовані програмні платформи, призначені для організації та виконання розподілених обчислень над великими даними. Найбільш відомими та широко використовуваними серед таких платформ є Hadoop та Spark (обидві входять до складу проектів фонду Apache Software Foundation і тому їх також часто називають Apache Hadoop та Apache Spark [25, 26]). У R є кілька пакетів (зокрема, sparklyr), що надають зручний інтерфейс для роботи з цими платформами.

HDFS - (Hadoop Distributed File System) - файлова система, призначена для зберігання файлів великих розмірів, побічно розподілених між вузлами обчислювального кластера. Усі блоки HDFS (крім останнього блоку файлу) мають однаковий розмір, і кожен блок може бути розміщений на декількох вузлах, розмір блоку і коефіцієнт реплікації (кількість вузлів, на яких повинен бути розміщений кожен блок) визначаються в налаштуваннях на рівні файлу. Завдяки реплікації забезпечується стійкість розподіленої системи до відмов окремих вузлів. Файли HDFS можуть бути записані лише один раз

(модифікація не підтримується), а запис у файл одночасно може вести тільки один процес.

Хоча платформа Hadoop дозволила багатьом компаніям успішно застосовувати парадигму MapReduce для розподілених обчислень над величезними обсягами даних, щоразу при виникненні нового завдання потрібно було написання нового коду для операцій map та reduce, що було незручно та трудомістко. Для вирішення цієї проблеми у 2008 р. інженери з Facebook створили Hive – систему управління базами даних на основі Hadoop. Головною особливістю Hive стала підтримка SQL-подібних запитів до даних, що зберігаються в HDFS (цей діалект SQL отримав назву Hive Query Language, HQL).

У 2009 р. у Каліфорнійському університеті в Берклі було запущено дослідницький проект Spark з метою підвищення ефективності розподілених обчислень методом MapReduce та створити універсальну платформу для таких обчислень. У 2010 році Spark був опублікований як проект з відкритим кодом, а в 2013 році передано фонду Apache Software Foundation. Сьогодні Spark є однією з найбільш широко використовуваних платформ для роботи з великими даними та характеризується такими особливостями: неперевершена швидкодія, що досягається за рахунок виконання обчислень в оперативній пам'яті великої кількості комп'ютерів, об'єднаних в один кластер, а також завдяки ефективним протоколам передачі даних через мережу; універсальність: Spark підтримує багато технологій кластерних обчислень і має кілька бібліотек-надбудов для вирішення поширених аналітичних завдань, включаючи Spark SQL (SQL-подібні запити до даних), MLlib (алгоритми машинного навчання), GraphX (аналіз графів) та Spark Streaming (обробка потокових) даних).

Пакет sparklyr надає зручний інтерфейс для роботи зі Spark-кластерами із середовища R. Зокрема, за його допомогою можна: встановлювати з'єднання із кластером; виконувати звичайні операції перетворення, фільтрації та агрегування даних з використанням синтаксису dplyr; будувати передбачувальні моделі з використанням алгоритмів машинного навчання, реалізованих у бібліотеці MLlib для Spark; працювати з іншими R-пакетами, які

використовують Spark для виконання розподілених обчислень (наприклад, rsparkling для роботи з фреймворком h2o).

Віртуальна машина Java (Java Virtual Machine; JVM) - віртуальна машина для виконання байт-коду Java [23]. JVM доступна для всіх основних сучасних платформ, тому програми, що скомпільовані у Java байткод можуть працювати всюди.

Пакет nycflights13 – це пакет, що містить інформацію про всі рейси, які вирушили з Нью-Йорка (наприклад, EWR, JFK і LGA) до пунктів призначення в Сполучених Штатах, Пуерто-Ріко та на Американських Віргінських островах) у 2013 році: загалом 336 776 рейсів. Щоб зрозуміти, що спричиняє затримки, він також містить ряд інших корисних наборів даних. Цей пакет містить такі таблиці даних [29]:

?flights: усі рейси, які вирушили з Нью-Йорка у 2013 році;

?weather: погодинні метеорологічні дані для кожного аеропорту;

?planes: інформація про конструкцію кожного літака;

?airports: назви та розташування аеропортів;

?airlines: переклад між двома буквами кодів перевізника та імен.

MongoDB – це документо-орієнтована система керування базами даних з відкритим вихідним кодом. MongoDB є безкоштовним і кросплатформним продуктом. Деякі функції, які підтримує MongoDB: спеціальні запити, індексація тиражування, балансування навантаження, зберігання файлів, агрегація, виконання JavaScript на стороні сервера, обмежені колекції, транзакції [27].

Щоб завантажити дані з бази даних MongoDB у фреймворк R, використовується бібліотека MongoLite. Це високорівневий, високопродуктивний клієнт MongoDB на основі libmongoc та jsonlite. Включає підтримку агрегації, індексування, map-reduce, потокового передавання, SSL-шифрування та SASL-автентифікації.

Наприклад:

```
# Use MongoLite library:
#install.packages("mongolite")
```

library(jsonlite)

library(mongolite)

Бінарні пакунки mongolite для OS-X або Windows можна встановити безпосередньо з CRAN:

install.packages("mongolite").

Допоміжні джерела інформації:

1. The Java® Virtual Machine Specification. Chapter 1. Introduction [Електронний ресурс]/ - Режим доступу: https://docs.oracle.com/javase/specs/jvms/se17/html/jvms-1.html (дата звернення: 10.11.2023)

2. 4 Types of Big Data Technologies (+ Management Tools)// Coursera, 16.06.2023 [Електронний ресурс] – Режим доступу: https://www.coursera.org/articles/big-data-technologies (дата звернення: 16.10.2023)

3. R and Hadoop Integration – Enhance your skills with different methods! [Електронний ресурс] – Режим доступу: https://data-flair.training/blogs/r-hadoop-integration/ (дата звернення: 16.10.2023).

4. Hadoop vs. Spark: What's the Difference? // IBM, 05.2021 [Електронний ресурс] – Режим доступу: https://www.ibm.com/blog/hadoop-vs-spark/ (дата звернення: 17.10.2023).

5. Getting Started with MongoDB [Електронний ресурс] – Режим доступу: https://data-flair.training/blogs/mongodb-tutorials-home/ (дата звернення: 16.10.2023)

6. Connect to MongoDB Database in R [Електронний ресурс] – Режим доступу: https://www.youtube.com/watch?v=JBEKJflNV2g (дата звернення: 16.10.2023)

7. nycflights13 [Електронний ресурс] – Режим доступу: https://github.com/tidyverse/nycflights13 (дата звернення: 16.10.2023).

Контрольні запитання до лабораторної роботи 7:

1. Що представляє собою HDFS?

2. Якими особливостями характеризується платформа Spark сьогодні?

3. Що спільного у Hadoop та Spark?

4. Які можливості надає пакет sparklyr в R?

5. Що представляє собою JVM?

6. Що представляє собою пакет nycflights13?

7. Що представляє собою MongoDB?

8. Для чого застосовують бібліотеку MongoLite?

Лабораторна робота №8

Тема: Використання різних інструментів обробки великих даних

Мета: Навчитись застосовувати різні інструменти при роботі з великими даними

Завдання:

1. Виберіть два будь-яких кейса [30]. Опишіть вхідні дані, стек моделей і технології, які можна використовувати для вирішення обраних кейсів.

2. Наведіть ілюстративні приклади з використанням R. Зробіть висновки.

Теоретичні відомості:

На сучасну індустрію впливають процеси, пов’язані з великими даними. Оскільки великі дані продовжують проникати в повсякденне життя, кількість різних компаній, які впроваджують BigData, продовжує зростати.

Розглядаються різноманітні випадки використання великих даних, коли індустрія використовує різні інструменти BigData (такі як Hadoop, Spark, Flink тощо) для вирішення конкретних проблем [31].

Приклади використання Apache Flink в основному зосереджені на аналітиці в реальному часі, в той час як приклади використання Apache Spark зосереджені на складних ітеративних реалізаціях алгоритмів машинного навчання.

Приклади використання Apache Hadoop зосереджені на ефективній обробці величезних обсягів даних.

Розглядаються 6 тематичних досліджень великих даних [30], що демонструють поміч BigData у експоненціальному зростанні на ринку.

Топ-5 кейсів з використанням великих даних. Приклади:

1. Кейс з великих даних – Walmart.

Walmart - найбільша роздрібна мережа у світі та найбільша компанія у світі за доходами, з більш ніж 2 млн. працівників та 20000 магазинів у 28 країнах. Вона почала використовувати аналітику великих даних задовго до того, як з'явилося саме слово "великі дані".

Walmart використовує Data Mining для виявлення закономірностей, які можна використовувати для надання користувачеві рекомендацій щодо продуктів, на основі яких були об'єднані товари.

Застосовуючи ефективний Data Mining, WalMart збільшив коефіцієнт конверсії своїх клієнтів. Він прискорює аналіз великих даних, щоб забезпечити найкращі у своєму класі технології електронної комерції з мотивом забезпечити чудовий клієнтський досвід.

Основною метою зберігання великих даних у Walmart є оптимізація купівельного досвіду клієнтів, коли вони перебувають у магазині Walmart.

Рішення на основі великих даних у Walmart розробляються з метою редизайну глобальних веб-сайтів і створення інноваційних додатків, щоб адаптувати досвід покупок для клієнтів, одночасно підвищуючи ефективність логістики.

Технології Hadoop та NoSQL використовуються для надання внутрішнім клієнтам доступу до даних у режимі реального часу, зібраних з різних джерел та централізованих для ефективного використання.

2. Приклад використання великих даних – Uber.

Uber - це перший вибір для людей по всьому світу, коли вони думають про перевезення людей та доставку. Він використовує персональні дані користувачів, щоб уважно стежити за тим, які функції сервісу використовуються найчастіше, аналізувати моделі використання і визначати, на чому слід зосередити увагу в наданні послуг.

Uber орієнтується на попит і пропозицію послуг, через що змінюються ціни на послуги, що надаються. Тому одним з найбільших напрямків використання даних в Uber є ціноутворення. Наприклад, якщо ви запізнюєтеся на зустріч і замовляєте таксі в людному місці, ви повинні бути готові заплатити вдвічі більше.

Наприклад, напередодні Нового року ціна за проїзд однієї милі може зрости від 200 до 1000 гривень. У короткостроковій перспективі стрибкоподібне ціноутворення впливає на рівень попиту, тоді як у довгостроковій перспективі воно може стати ключем до утримання або втрати

клієнтів. Алгоритми машинного навчання допомагають визначити, де попит є найвищим.

3. Приклад використання великих даних – Netflix.

Це найулюбленіша американська розважальна компанія, що спеціалізується на потоковому відео на вимогу для своїх клієнтів.

За допомогою великих даних Netflix вирішила передбачити, що саме сподобається її клієнтам. Таким чином, аналітика великих даних є паливом, яке запускає "двигун рекомендацій", призначений для цієї мети. Зовсім недавно Netflix почав позиціонувати себе як творець контенту, а не лише як спосіб його розповсюдження.

Не дивно, що ця стратегія значною мірою ґрунтується на даних. Механізми рекомендацій Netflix та рішення щодо нового контенту ґрунтуються на даних про те, які фільми дивляться клієнти, як часто відтворення зупинялося, які оцінки ставлять тощо. Структура даних компанії включає Hadoop, Hive і Pig, а також багато інших традиційних засобів бізнес-аналітики.

Досвід Netflix показує, що точно знати, чого хочуть клієнти, легко, якщо компанії не обмежуються припущеннями, а приймають рішення з BigData.

4. Приклад використання великих даних – eBay.

Великим технічним викликом для eBay, як для бізнесу, що оперує великими обсягами даних, є використання системи, яка може швидко аналізувати дані та діяти на них по мірі їх надходження (потокові дані). Існує багато методів, що швидко розвиваються, для підтримки аналізу потокових даних.

eBay працює з кількома інструментами, серед яких Apache Spark, Storm, Kafka. Це дозволяє аналітикам компанії шукати інформаційні теги, які були пов'язані з даними (метадані), і робити їх доступними для якомога більшої кількості людей з належним рівнем безпеки та дозволів (управління даними).

Компанія знаходиться на передньому краї використання рішень для роботи з великими даними і активно ділиться своїми знаннями зі спільнотою розробників відкритого програмного забезпечення.

5. Приклад використання великих даних - Procter & Gamble.

Procter & Gamble, продукцією якої ми всі користуємося 2-3 рази на день, - 179-річна компанія. Ця геніальна компанія визнала потенціал великих даних і почала використовувати їх у своїх бізнес-підрозділах по всьому світу. P&G робить сильний акцент на використанні великих даних для прийняття кращих, розумніших бізнес-рішень у режимі реального часу.

Організація Global Business Services розробила інструменти, системи та процеси, які надають менеджерам прямий доступ до найновіших даних та передової аналітики. Тому P&G, будучи найстарішою компанією, все ще утримує велику частку на ринку, незважаючи на те, що на ньому є багато компаній, що розвиваються.

Великі дані прогнозують невизначеність.

Революційне дослідження, проведене в Бангладеш, показало, що використання даних з мобільних телефонних мереж для відстеження пересування людей по всій країні допомагає передбачити, де можуть виникнути спалахи таких захворювань, як малярія, що дозволяє органам охорони здоров'я вживати превентивних заходів.

Щороку малярія забирає понад 400 000 життів у світі, і більшість з них - діти.

Різні типи даних, включаючи інформацію, надану Міністерством охорони здоров'я Бангладеш, використовуються для створення карт ризику із зазначенням ймовірних місць спалахів малярії, щоб місцеві органи охорони здоров'я були попереджені про необхідність вжити превентивних заходів, включаючи розпилення інсектицидів і створення запасів надкроватних сіток і ліків для захисту населення від цієї хвороби [30].

Кейси використання великих даних і додатків для роботи з великими даними в різних сферах:

1. Виявлення шахрайства з кредитними картками [31];

Транзакція за кредитною карткою займає не більше 2-4 секунд. Тому компаніям потрібне інноваційне рішення, щоб за цей короткий час виявити транзакції, які можуть виглядати як шахрайство, і таким чином захистити своїх клієнтів від того, щоб вони не стали його жертвами.

Для обробки потоків даних нам потрібні потокові рушії, такі як Apache Flink. Потоковий движок може споживати потоки даних у реальному часі з дуже високою ефективністю і обробляти дані з низькою латентністю (без затримок)

2. Аналіз настроїв [31];

Аналіз настроїв дає змогу розкрити зміст соціальних даних. Основним завданням аналізу настроїв є класифікація полярності тексту на рівні документа, речення або властивості/аспекту - чи є висловлена думка в документі, реченні або властивості/аспекті сутності позитивною, негативною або нейтральною.

Розширена, "позаполярна" класифікація настроїв розглядає, наприклад, такі емоційні стани, як "злий", "сумний" і "щасливий".

При аналізі настроїв мова обробляється для виявлення та розуміння почуттів і ставлення споживачів до брендів або тем в онлайн-розмовах, тобто, що вони думають про певний продукт або послугу, чи задоволені вони нею, чи ні тощо.

За допомогою Hadoop є можливим аналізувати розмови в Twitter, Facebook та інших соціальних мережах для отримання даних про настрої щодо вас і ваших конкурентів, і використовувати їх для прийняття цільових рішень в режимі реального часу, які збільшують частку ринку.

3. Обробка даних (роздрібна торгівля)[31];

Клієнт звернувся до команди Big Data однієї з компаній зі своєю проблемою витрачання часу на звіт з надією отримати краще рішення. Коли команда Big Data почала працювати над його проблемою і повернулася з рішенням, клієнт був вражений і не міг повірити, що звіт, який він отримував за 10 годин, тепер можна отримати всього за 10 хвилин, використовуючи Big Data і Hadoop.

4. Orbitz.com [31];

Orbitz - провідна туристична компанія, яка використовує новітні технології, щоб трансформувати спосіб планування подорожей клієнтів по

всьому світу. Компанія управляє сайтами для планування подорожей Orbitz, Ebookers та CheapTickets.

Щодня компанія генерує 1,5 мільйона запитів на пошук авіаквитків та 1 млн. запитів на пошук готелів, а обсяг журналу даних, який генерується в результаті цієї діяльності, становить приблизно 500 ГБ. Свіжі журнали зберігаються лише кілька днів через дорожнечу зберігання даних.

Обробка таких величезних обсягів даних та їх зберігання за допомогою звичайної інфраструктури зберігання та аналізу даних ставала з часом все дорожчою і забирала все більше часу.

Для вирішення проблеми були використані HDFS, Map Reduce і Hive, і були отримані просто приголомшливі результати. Кластер Hadoop забезпечив дуже економічно ефективний спосіб зберігання величезних обсягів необроблених логів. Дані очищаються, аналізуються та запускаються алгоритми машинного навчання.

5. Sears Holding [31];

Sears Holding використовує Hadoop для персоналізації маркетингових кампаній.

Проблема полягала в тому, що застарілі системи стали нездатними аналізувати великі обсяги даних для персоналізації маркетингових кампаній та кампаній лояльності.

Вони хотіли персоналізувати маркетингові кампанії, купони та пропозиції для кожного окремого клієнта, але наші застарілі системи були нездатні підтримувати те, що призводило до зниження їхніх прибутків.

З'явилася найсучасніша реалізація Apache Hadoop, високомасштабної платформи обробки даних з відкритим вихідним кодом, яка є рушійною силою тренду великих даних.

Для певних сценаріїв онлайн і мобільної комерції Sears тепер може виконувати щоденний аналіз. Інтерактивні звіти можуть бути розроблені за 3 дні замість 6-12 тижнів за допомогою цього методу.

Цей крок заощадив мільйони доларів на мейнфреймах і СКБД і дозволив Sears отримати кращу продуктивність.

Допоміжні джерела інформації:

1. https://data-flair.training/blogs/big-data-case-studies/
2. https://data-flair.training/blogs/big-data-use-cases-case-studies-hadoop-spark-flink/

Контрольні запитання до лабораторної роботи 8:

1. Який вплив має BigData на сучасну індустрію?
2. Які інструменти BigData застосовуються в сучасній індустрії?
3. Які провідні компанії застосовують великі дані для зростання?
4. Які проблеми вирішують інструменти великих даних?

Додаток 1

Приклад звіту про виконаня лабораторної роботи

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № _

з навчальної дисципліни “Big Data”

на тему: Знайомство з мовою R

Виконав: студент/ка

групи КІ/КН-23М

Морозко Т. Г.

Перевірив: викладач

Константинова Л.В.

Кропивницький 2024

МЕТА: Ознайомитись з основними характеристиками мови R, навчитись встановлювати R та працювати в середовищі RStudio.

Завдання:

1. Встановити та запустити програму R.
2. Налаштувати робочу директорію за допомогою функції setwd ().
3. Згенерувати вектор X ~ U (0, 1) і побудувати гістограму.
4. Зберегти результати моделювання в текстовий файл (* .txt, * .csv), гистограмму в графічний файл (* .jpg, * .svg) з використанням функцій запису в файл.
5. Ознайомитись з основними структурами даних numeric (), matrix (), data.frame (), list ().
6. Ознайомитись з довідковою системою.
7. Установка і завантаження додаткових пакетів за допомогою функцій install.packages () ilibrary ().

Хід роботи:

1. Для встановлення R було виконано наступні дії: ……..(опис з підтвердженням скрінами)

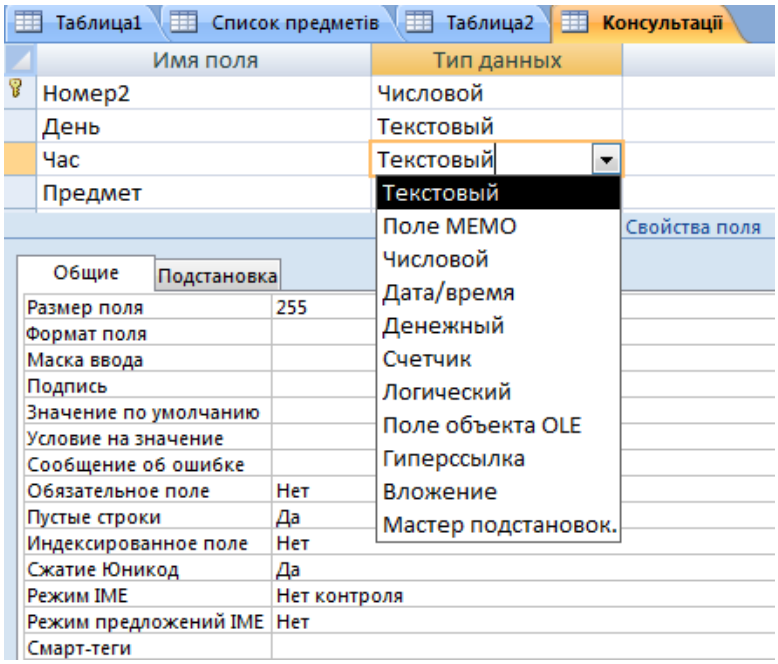


Рисунок 1 – Головне вікно програми

2. Для налаштування було виконано ….. , що представлено на рисунку 2

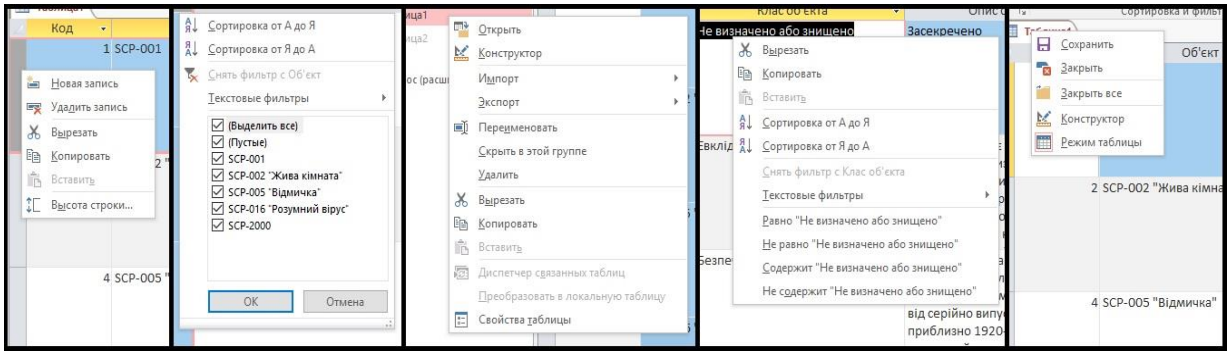


Рисунок 2 – Зміна директорії

3. ….

….

7. ….

Контрольні питання до лабораторної роботи 1 і відповіді:

1. Що розуміють під поняттям дані?
2. Яким чином оголошують змінні в R.
3. За допомогою якої функції можна налаштувати робочу директорію?
4. За допомогою якої функції можна перевірити яка встановлена робоча директорія?
4. Що представляє собою вектор?
5. Як визначити кількість елементів вектору?
6. Як побудувати гістограму в R?
7. Як здійснити пошук у довідці записів про сортування?
8. Перелічіть основні структури даних в R?
9. Як здійснити встановлення та завантаження додаткових пакетів?

…..

(відповіді)…..

……

Висновок:

У процесі роботи я ознайомився на практиці з ……. Навчився створювати …... Навчився працювати з даними …….(свої висновки)

Рекомендовані джерела інформації

1. Tilman M. Davies The Book of R : a first course in programming and statistics. No Starch Press, Inc. San Francisco, 2016 [Електронний ресурс]/ - Режим доступу: https://web.itu.edu.tr/~tokerem/The_Book_of_R.pdf (дата звернення: 7.11.2023).

2. Лекція 2. Робота з мовою R [Електронний ресурс]/ - Режим доступу: https://rstudio-pubs-static.s3.amazonaws.com/378105_599bcd2892bf46498a6371290149267d.html (дата звернення: 6.11.2023).

3. Довідка з R // Статистика й теорія ймовірностей [Електронний ресурс]/ - Режим доступу: https://tvimc.jimdofree.com/%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D1%87%D0%BD%D1%96-%D0%BC%D0%B0%D1%82%D0%B5%D1%80%D1%96%D0%B0%D0%BB%D0%B8/%D0%B4%D0%BE%D0%B2%D1%96%D0%B4%D0%BA%D0%B0-%D0%B7-r/ (дата звернення: 20.11.2023).

4. R Introduction // W3Schools [Електронний ресурс] – Режим доступу : https://www.w3schools.com/r/r_intro.asp (дата звернення: 17.10.2023).

5. Quickly reading very large tables as dataframes // Stackoverflow [Електронний ресурс] – Режим доступу : https://stackoverflow.com/questions/1727772/quickly-reading-very-large-tables-asdataframes/ (дата звернення: 10.10.2023).

6. Big Data Analytics - Introduction to SQL // Tutorials Point [Електронний ресурс] – Режим доступу : https://www.tutorialspoint.com/big_data_analytics/introduction_to_sql.htm (дата звернення: 10.11.2023).
7. Лекція 6. Робота з функціями [Електронний ресурс] – Режим доступу : https://rstudio-pubs-static.s3.amazonaws.com/378109_3e5dfcb6c47a44fbbb8b420902cd342f.html (дата звернення: 11.11.2023).
8. Outlier detection and treatment with R. // R-bloggers [Електронний ресурс] – Режим доступу : https://www.r-bloggers.com/2016/12/outlier-detection-and-treatment-with-r/ (дата звернення: 15.11.2023).
9. Bidyut Ghosh Multicollinearity in R // DataSciencePlus.com, Sep 29, 2017 [Електронний ресурс] – Режим доступу : https://datascienceplus.com/multicollinearity-in-r/ (дата звернення: 21.11.2023).
10. 7 train Models By Tag [Електронний ресурс] – Режим доступу : https://topepo.github.io/caret/train-models-by-tag.html#implicit-feature-selection (дата звернення: 28.11.2023).
11. Feature Selection with FSelector Package [Електронний ресурс] – Режим доступу : https://miningthedetails.com/blog/r/fselector/ (дата звернення: 28.11.2023).
12. Miron B. Kursa, Witold R. Rudnicki. Feature Selection with the Boruta Package // Journal of Statistical Software, September 2010, Volume 36, Issue 11. [Електронний ресурс] – Режим доступу : https://www.jstatsoft.org/article/view/v036i11/v36i11.pdf (дата звернення: 28.11.2023).

13. Feature Selection in R with the Boruta R Package, 2018// Datacamp [Електронний ресурс] – Режим доступу : https://www.datacamp.com/community/tutorials/feature-selection-R-boruta (дата звернення: 27.11.2023).
14. Ron Kohavi, George H. John. Wrappers for feature subset selection // [Електронний ресурс] – Режим доступу : http://ai.stanford.edu/~ronnyk/wrappersPrint.pdf (дата звернення: 27.11.2023).
15. Розвідувальний аналіз структури даних // Бібліотека підручників та статтей Posibniki (2022) [Електронний ресурс] – Режим доступу : https://posibniki.com.ua/post-rozviduvalnii-analiz-strukturi-danih-instrumenti-i-strategiyi (дата звернення: 28.11.2023).
16. Classification using k-Nearest Neighbors in R // proft.me 22.01.2017 [Електронний ресурс] – Режим доступу : https://en.proft.me/2017/01/22/classification-using-k-nearest-neighbors-r/ (дата звернення: 28.11.2023).
17. Olvera-López, José& Carrasco-Ochoa, Jesús& Martínez-Trinidad, JoséFrancisco&Kittler, Josef. (2010). A review of instance selection methods. Artif. Intell. Rev. 34. 133-143.10.1007/s10462-010-9165-y. [Електронний ресурс] – Режим доступу : https://m.moam.info/a-review-of-instance-selection-methods-softcomputing-and-_5b054f698ead0ed4758b4586.html (дата звернення: 29.11.2023).
18. Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang ·Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu,·Zhi-Hua Zhou, Michael Steinbach, David J. Hand, Dan Steinberg. Top 10 algorithms in data mining [Електронний ресурс] – Режим доступу : http://www.cs.umd.edu/~samir/498/10Algorithms-08.pdf (дата звернення: 29.11.2023).
19. The Comprehensive R Archive Network // CRAN [Електронний ресурс] – Режим доступу : https://cran.r-project.org/ (дата звернення: 29.11.2023).
20. R/discretize.R // rdrr.io [Електронний ресурс] – Режим доступу : https://rdrr.io/cran/arules/src/R/discretize.R (дата звернення: 29.11.2023).

21. [Електронний ресурс] – Режим доступу : https://www.rdocumentation.org/packages/discretization/versions/1.0-1.1/topics/caim (дата звернення: 29.11.2023).
22. Kurgan, L. and Cios, K.J., 2004. CAIM Discretization Algorithm. IEEE Transactions on Knowledge and Data Engineering, 16(2):145-153
23. The Java Virtual Machine Specification. Chapter 1. Introduction [Електронний ресурс] - Режим доступу: https://docs.oracle.com/javase/specs/jvms/se17/html/jvms-1.html (дата звернення: 10.11.2023).
24. 4 Types of Big Data Technologies (+ Management Tools)// Coursera, 16.06.2023 [Електронний ресурс] – Режим доступу: https://www.coursera.org/articles/big-data-technologies (дата звернення: 16.10.2023)
25. R and Hadoop Integration – Enhance your skills with different methods! [Електронний ресурс] – Режим доступу: https://data-flair.training/blogs/r-hadoop-integration/ (дата звернення: 16.10.2023).
26. Hadoop vs. Spark: What's the Difference? // IBM, 05.2021 [Електронний ресурс] – Режим доступу: https://www.ibm.com/blog/hadoop-vs-spark/ (дата звернення: 17.10.2023).
27. Getting Started with MongoDB [Електронний ресурс] – Режим доступу: https://data-flair.training/blogs/mongodb-tutorials-home/ (дата звернення: 16.10.2023).
28. Connect to MongoDB Database in R [Електронний ресурс] – Режим доступу: https://www.youtube.com/watch?v=JBEKJflNV2g (дата звернення: 16.10.2023).
29. nycflights13 [Електронний ресурс] – Режим доступу: https://github.com/tidyverse/nycflights13 (дата звернення: 16.10.2023).
30. 5 Big Data Case Studies – How big companies use Big Data // DataFlair [Електронний ресурс] – Режим доступу: https://data-flair.training/blogs/big-data-case-studies/ (дата звернення: 20.11.2023).

31. Big Data Use Cases – Hadoop, Spark and Flink Case Studies // DataFlair [Електронний ресурс] – Режим доступу: https://data-flair.training/blogs/big-data-use-cases-case-studies-hadoop-spark-flink/ (дата звернення: 20.11.2023).
32. Top Big Data Technologies in 2023: How They Can Benefit Your Business// DOIT Software, 18.08.2023 [Електронний ресурс] – Режим доступу: https://doit.software/blog/big-data-technologies# (дата звернення: 15.10.2023).
33. Data Mining Tools// RapidMiner [Електронний ресурс] – Режим доступу: https://rapidminer.com/glossary/data-mining-tools/. (дата звернення: 17.10.2023)
34. Технології оброблення великих даних: комп'ютерний практикум з дисципліни «Технології оброблення великих даних» [Електронний ресурс] : навч. посіб. для студ. спеціальності 121 «Інженерія програмного забезпечення» (освітня програма «Інженерія програмного забезпечення мультимедійних та інформаційно-пошукових систем»)/ Л.М. Олещенко; КПІ ім. Ігоря Сікорського. – Електронні текстові дані. – Київ: КПІ ім. Ігоря Сікорського, 2021. – 85 с.