Міністерство освіти і науки України
Національний університет «Запорізька політехніка»

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт
з дисципліни

«СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ»

для студентів спеціальностей
121 Інженерія програмного забезпечення
122 Комп'ютерні науки
освітнього рівня "бакалавр"
усіх форм навчання

2023

Методичні вказівки до виконання лабораторних і самостійних робіт з дисципліни «Системи штучного інтелекту» для студентів спеціальностей 121 Інженерія програмного забезпечення, 122 Комп'ютерні науки освітнього рівня "бакалавр" усіх форм навчання / Уклад.: С.О. Субботін. – Запоріжжя: НУ «Запорізька політехніка», 2023. – 24 с.

Автори: Сергій Олександрович Субботін,
завідувач кафедри програмних засобів,
д.т.н., професор

Рецензент: А.О. Олійник, д.т.н., проф.

Відповідальний
за випуск: С.О. Субботін, зав. каф. програмних засобів

Затверджено
на засіданні кафедри
програмних засобів

Протокол № 12
від 09.06.2023 р.

ЗМІСТ

ЗАГАЛЬНІ ПОЛОЖЕННЯ

Дане видання призначене для вивчення та практичного освоєння студентами усіх форм навчання основ створення систем штучного інтелекту.

Відповідно до графіка студенти перед виконанням лабораторної або самостійної роботи повинні ознайомитися з конспектом лекцій та рекомендованою літературою.

Для одержання заліку по кожній роботі студент здає викладачу цілком оформлений звіт, а також комплект файлів, перевірений на відсутність вірусів, з текстами розроблених програм, виконуваними файлами програм, файлами даних і текстом звіту.

Звіт (приклад оформлення - див. додаток А) має містити:

- титульний аркуш (на ньому вказують назву міністерства, назву університету, назву кафедри, номер, вид і тему роботи, виконавця та особу, що приймає звіт, рік);
- мету, варіант і завдання роботи;
- лаконічний опис теоретичних відомостей;
- текст програми, що обов'язково містить коментарі;
- вхідні та вихідні дані програми;
- змістовний аналіз отриманих результатів та висновки.

Звіт виконують на білому папері формату А4 (210 х 297 мм). Текст розміщують тільки з однієї сторони листа. Поля сторінки з усіх боків – 20 мм. Аркуші скріплюють за допомогою канцелярських скріпок. Для набору тексту звіту використовують редактор MS Word 97: шрифт Times New Roman, 12 пунктів. Міжрядковий інтервал: полуторний – для тексту звіту, одинарний – для листингів програм, таблиць і роздруківок даних.

Під час співбесіди студент повинний виявити знання про мету роботи, по теоретичному матеріалу, про методи виконання кожного етапу роботи, по змісту основних розділів розробленого звіту з демонстрацією результатів на конкретних прикладах. Студент повинний вміти правильно аналізувати отримані результати. Для самоперевірки при підготовці до виконання і здачі роботи студент повинний відповісти на контрольні питання, приведені наприкінці опису відповідної роботи. Підсумкову оцінку за лабораторний практикум студент одержує після виконання і здачі останньої роботи.

1 Лабораторна робота № 1

РОЗПІЗНАВАННЯ ОБРАЗІВ НА ОСНОВІ МЕТРИЧНОЇ КЛАСИФІКАЦІЇ

Мета роботи: вивчити та засвоїти на практиці метричні методи розпізнавання образів у просторі ознак, навчитися створювати програмні засоби, що реалізують методи метричної класифікації, порівняти результати роботи метричної класифікації з іншими видами класифікації.

Завдання до роботи

Частина 1.

1.1. Ознайомитися з конспектом лекцій та рекомендованою літературою. На мові програмування Python написати програму, що реалізує процедури для навчання та емуляції за методом еталонів.

1.2. Згідно з номером студента за журналом для відповідного номера варіанта V сформувати навчаючу вибірку <x,y> обсягом S екземплярів x^s, s=1, 2, …,S, що характеризуються N ознаками x^s_j, j=1,2, …,N, та зіставити кожному екземпляру значення цільової ознаки y^s:

$$x^s_j = \begin{cases} jV - 0{,}1s,\ j = 1,5,9,..., \\ 0{,}01jV^{-1} + 0{,}3s,\ j = 2,4,6,..., \\ jrand,\ j = 3,7,11,...; \end{cases} \qquad y^s = \begin{cases} 0, (x^s_1)^2 + (x^s_2)^2 < V^2 + 0{,}04S^2, \\ 1, (x^2_1)^2 + (x^2_2)^2 \geq V^2 + 0{,}04S^2; \end{cases}$$

$$S = \begin{cases} 10V, V < 10, \\ 5V, 10 \leq V < 20, \\ 3V, V \geq 20; \end{cases} \qquad N = \begin{cases} 5V, V < 7, \\ 4V, 7 \leq V < 10, \\ 3V, 10 \leq V < 20, \\ 2V, V \geq 20; \end{cases}$$

де rand - випадкове число в діапазоні [0, 1].

1.3. Виконати нормування навчаючої вибірки даних.

1.4. На основі пронормованої вибірки побудувати розпізнаючу модель за методом еталонів, тобто визначити координати центрів класів (еталонів) у просторі ознак C^q_j, де q - номер класу, j - номер ознаки. Зафіксувати час навчання.

1.5. На основі побудованої моделі для екземплярів навчаючої вибірки виконати розпізнавання, тобто визначити розрахункові номери класів y^{s*}. Зафіксувати час розпізнавання.

1.6. Обчислити помилку розпізнавання, визначити ймовірність прийняття правильного рішення та ймовірність прийняття помилкового рішення для побудованої моделі.

1.7. У попередньо сформованій вибірці залишити тільки одну ознаку, номер якої у попередній вибірці дорівнює V, а також у центрів еталонів класів залишити тільки V-ту координату C^q_V.

1.8. Для нової вибірки та нових еталонів виконати розпізнавання, тобто визначити розрахункові номери класів y^{s*}. Зафіксувати час розпізнавання.

1.9. Обчислити помилку розпізнавання, визначити ймовірність прийняття правильного рішення та ймовірність прийняття помилкового рішення для нової моделі.

1.10. Порівняти результати проведених експериментів, зробити висновки щодо впливу параметрів навчаючої вибірки на характеристики процесів навчання та розпізнавання.

Частина 2. Класифікація вибірки даних «Titanic train» за допомогою алгоритмів пакету «Sklearn»

2.1. Описати вибірку, надати короткий опис наявним в вибірці ознакам, проаналізувати їх, визначити малозначущі і видалити їх (за наявності).

2.2. Дослідити наявність пропущених даних, їх кількість. За потреби усунути нестачу будь-яким прийнятним методом. Обґрунтувати вибір.

2.3. Виконати розпізнавання наступними алгоритмами з пакету «Sklearn» і порівняти їх з результатами метричної класифікації (методом найближчих сусідів):

- методом найближчих сусідів (KNeighborsClassifier);
- випадковий ліс (RandomForestClassifier);
- метод логістичної регресії (LogisticRegression);
- метод опорних векторів (SVC).

4. Результати розпізнавання відобразити у вигляді графіку, проаналізувати їх, зробити припущення як їх можна покращити.

Зміст звіту

1. Мета роботи.

2. Короткі теоретичні відомості до роботи. У звіті не слід наводити докладне викладення теоретичного матеріалу, необхідно виділити лише найголовніші ідеї, формули, що необхідні для пояснення суті методу, моделі тощо.

3. Текст розроблених програм.

4. Сформована навчаюча вибірка <x,y>.

5. Побудована модель - координати еталонів (центрів зосередження екземплярів) класів. Час навчання.

6. Результати розпізнавання: розраховані значення номеру класу для екземплярів y^{s*}, помилка розпізнавання, імовірності прийняття правильного та помилкового рішень, час розпізнавання.

7. Навчаюча вибірка із однією ознакою.

8. Еталони класів з однією ознакою.

9. Результати розпізнавання скороченої вибірки за скороченими еталонами: розраховані значення номеру класу для екземплярів y^{s*}, помилка розпізнавання, імовірності прийняття правильного та помилкового рішень, час розпізнавання.

10. Стислий опис вибірки «Titanic train» і її ознак.

11. Результати з виявлення і усунення (за наявності) відсутніх даних. Обґрунтування обраного методу виправлення.

12. Результати розпізнавання алгоритмами з пакету «Sklearn», їх порівняння, припущення щодо можливого покращення результатів.

13. Висновки. У висновках треба проаналізувати результати роботи, а також лаконічно відповісти на контрольні питання.

Контрольні питання

1. Задача розпізнавання образів.
2. Основні поняття теорії розпізнавання образів.
3. Розбиття вихідної вибірки на навчаючу та тестову.
4. Навчання з учителем.
5. Методи метричної класифікації.
6. Призначення пакету «Sklearn».
7. Лінійна роздільність і лінійна нероздільність класів.

8. Помилка навчання / класифікації, час навчання / класифікації, цільова функція навчання.

9. Чи впливає кількість використаних ознак на швидкість навчання персептрона? Відповідь обґрунтуйте теоретично та доведіть експериментально.

10. Що таке генеральна сукупність, вибірка, екземпляр, ознака?

11. Вимоги до навчаючих вибірок даних.

12. Що таке репрезентативна вибірка даних?

13. Чи повинна навчальна вибірка бути репрезентативною?

14. Чи повинна тестова вибірка бути репрезентативною?

15. Чи впливає обсяг навчаючої вибірки на швидкість навчання?

16. Чи впливає репрезентативність навчаючої вибірки на точність класифікації екземплярів тестової вибірки?

17. Чи впливає репрезентативність тестової вибірки на точність класифікації екземплярів тестової вибірки?

18. Чи впливає репрезентативність тестової вибірки на точність навчання персептрона за навчаючої вибіркою?

19. Чи залежить якість навчання від якості та обсягу навчаючої вибірки?

2 Лабораторна робота № 2

САМООРГАНІЗАЦІЯ ТА НАВЧАННЯ БЕЗ УЧИТЕЛЯ. КЛАСТЕР-АНАЛІЗ

Мета роботи: вивчити та засвоїти на практиці методи кластер-аналізу та їх використання при зміні кількості ознак.

Завдання до роботи

1. Ознайомитися з конспектом лекцій та рекомендованою літературою.
2. Проаналізувати і описати вибірку «iris» з пакету «Sklearn.datasets», надати коротку характеристику наявним в вибірці ознакам.
3. Побудувати графіки розподілу значень від ознак (6 графіків, комбінації: (sepal length, sepal width), (sepal length, petal length) і т.д.), різними кольорами виділивши належність до цільового класу. Зробити припущення про результати кластеризації.
4. Дослідити вибірку нижчезазначеними методами. Для кожного методу: коротко пояснити суть алгоритму, розрахувати помилки роботи, проаналізувати їх.

4.1 Метод к–середніх (KMeans).

4.2 Метод ієрархічної кластеризації (Hierarchical clustering).

5. Врахувавши результати, отримані у лабораторній роботі 2 видалити частину ознак і повторити пункт пункт 4.
6. Порівняти отримані різними методами результати, зробити висновки.

Зміст звіту

1. Мета роботи.
2. Короткі теоретичні відомості до роботи.
3. Текст розробленої програми.
4. Стислий опис вибірки і її ознак.
5. Графічні і текстові відображення результатів роботи алгоритмів, їх порівняння.

6. Висновки. У висновках треба проаналізувати результати роботи, а також лаконічно відповісти на контрольні питання.

Контрольні питання

1. Задача кластер-аналізу.
2. Чіткий кластер-аналіз.
3. Нечіткий кластер-аналіз.
4. Використання кластер-аналізу при побудові систем розпізнавання образів.
5. Навчання з учителем.
6. Навчання без учителя.
7. Подібність кластер-аналізу і метричної класифікації.
8. Кластеризація методом к–середніх.
9. Лінійна роздільність і лінійна нерозділеність класів.
10. Чи впливає кількість використаних ознак на швидкість кластер-аналізу?
11. Що таке генеральна сукупність, вибірка, екземпляр, ознака?
12. Чи впливає обсяг навчаючої вибірки на швидкість кластер-аналізу?
13. Ієрархічна кластеризація.

3 Лабораторна робота № 3

МЕТОДИ ВІДБОРУ ОЗНАК ДЛЯ ПОБУДОВИ РОЗПІЗНАЮЧИХ МОДЕЛЕЙ

Мета роботи: вивчити та засвоїти на практиці методи оцінювання інформативності та відбору ознак, для побудови розпізнаючих моделей.

Завдання до роботи

1. Ознайомитися з конспектом лекцій та рекомендованою літературою.

2. Проаналізувати і описати вибірку «iris» з модулю «Sklearn.datasets», надати коротку характеристику наявним в вибірці ознакам.

3. Дослідити вибірку нижчезазначеними методами: Для кожного методу: коротко пояснити суть алгоритму, побудувати графічне відображення отриманих результату (коефіцієнтів), проаналізувати їх, налаштувати параметри методу так, що б алгоритм відсіяв хоча б 1 ознаку.

3.1 Метод виявлення ознак з низькою дисперсією (Variance Threshold).

3.2. Метод одновимірного відбору ознак (Univariate feature selection(UFS)).

3.3. Метод виявлення рекурсивних ознак (Recursive feature elimination(RFE)).

3.4. Метод оцінювання ознак на основі L1 алгоритму (L1-based feature selection).

3.5. Метод на основі дерева рішень (Каскадний метод, Tree-based feature selection).

4. Порівняти отримані різними методами результати, зробити висновки.

Зміст звіту

1. Мета роботи.
2. Короткі теоретичні відомості до роботи.
3. Текст розробленої програми.
4. Стислий опис вибірки і її ознак.
5. Графічні і текстові відображення результатів роботи алгоритмів, їх порівняння.
6. Висновки. У висновках треба проаналізувати результати роботи, а також лаконічно відповісти на контрольні питання.

Контрольні питання

1. Задача відбору ознак.
2. Критерії оцінювання інформативності ознак на основі евристичного, інформаційного, статистичного та імовірнісного підходів.
3. Виявлення ознак з низькою дисперсією, алгоритм, сутність.
4. Комбінована оцінка інформативності ознак.
5. Метод каскадного відбору ознак, його алгоритм, сутність.
6. Що таке генеральна сукупність, вибірка, екземпляр, ознака?
7. Вимоги до навчаючих вибірок даних.
8. Метод виявлення рекурсивних ознак, його алгоритм, сутність.
9. Що таке репрезентативна вибірка даних?
10. Який функціонал модулю «Sklearn.feature_selection»?

ЛІТЕРАТУРА

Основна

1. Субботін С.О. Інтелектуальні системи: навч. посіб [Електронний ресурс]. / С. О. Субботін, А. О. Олійник ; за ред. С. О. Субботіна. – Запоріжжя: ЗНТУ, 2014. – 219 с. – Режим доступу: http://eir.zntu.edu.ua/bitstream/123456789/2156/4/Subbotin_%d0%86ntelligent_%20systems.pdf
2. Троцько В.В. Методи штучного інтелекту: навчально-методичний і практичний посібник [Електронний ресурс]. – Київ: Університет економіки та права «КРОК», 2020 – 86 с. – Режим доступу: https://library.krok.edu.ua/media/library/category/navchalni-posibniki/trotsko_0001.pdf

Додаткова

3. Методи та системи штучного інтелекту: навч. посіб. [Електронний ресурс] / укл. Д.В. Лубко, С.В. Шаров. – Мелітополь: ФОП Однорог Т.В., 2019. – 264 с. – Режим доступу: http://www.tsatu.edu.ua/kn/wp-content/uploads/sites/16/knyha.-msshy-v-byblyoteku.pdf
4. Субботін С. О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей: монографія / С. О. Субботін, А. О. Олійник, О. О. Олійник ; під заг. ред. С. О. Субботіна. – Запоріжжя: ЗНТУ, 2009. – 375 с.
5. Математичні та програмні засоби для прийняття рішень, розпізнавання образів й інтелектуального діагностування: монографія / [С. О. Субботін, А. О. Олійник, Є. М. Федорченко та ін.]; під заг. ред. С. О. Субботіна. – Запоріжжя: НУ “Запорізька політехніка”, 2020. – 271 с.
6. Методи та системи штучного інтелекту: Навчальний посібник для студентів напряму підготовки 6.050101 «Комп'ютерні науки» [Електронний ресурс] / Уклад.: А.С. Савченко, О. О. Синельніков. – К.: НАУ, 2017. – 190 с. – Режим доступу: https://er.nau.edu.ua/bitstream/NAU/40676/1/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%20%D1%82%D0%B0%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8%20%D1%88%D1%82%D1%83%D1%87%D0%BD%D0%BE%D0%B3%D0%BE%20%D1%96%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%8

2%D1%83%20_%D0%9D%D0%B0%D0%B2%D1%87_%D0%BF%D0%BE%D1%81%D1%96%D0%B1%D0%BD.pdf

7. Субботін С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень: навчальний посібник / С. О. Субботін. – Запоріжжя: ЗНТУ, 2008. – 341 с.

8. Олійник А. О. Інтелектуальний аналіз даних: навчальний посібник / А. О. Олійник, С. О. Субботін, О.О. Олійник. – Запоріжжя: ЗНТУ, 2011. – 271 с.

9. Булгакова, О. С. Методи та системи штучного інтелекту: теорія та практика [Текст]: навчальний посібник / О. С. Булгакова, В. В. Зосімов, В. О. Поздєєв. – Херсон: ОЛДІ-ПЛЮС, 2020. – 356 с.

10.Литвин, В. В. Інтелектуальні системи [Текст]: підручник / В. В. Литвин, В. В. Пасічник, Ю. В. Яцишин ; за наук. ред. В. В. Пасічника. – Львів: Новий Світ-2000, 2020. – 406 с.

11. Зайченко Ю.П. Основи проектування інтелектуальних систем. / Ю.П. Зайченко Навчальний посібник.– К.: Слово, 2004.– 352 с.

12. Рідкокаша А.А.,Основи систем штучного інтелекту. / А.А. Рідкокаша, К.К. Голдер. Навчальний посібник.– Черкаси, "Відлуння–Плюс", 2002.–240 с.

Додаток А Рекомендації до роботи з Python

Python - високорівнева мова програмування, яка призначена для створення додатків різних типів. Це і веб-додатки, і гри, і настільні програми, і робота з базами даних. Досить велике поширення Python отримала в області машинного навчання і досліджень штучного інтелекту.

Встановлення і налаштування Python

Для створення програм на Python буде потрібно встановити інтерпретатор. Його рекомендовано завантажити з офіційного сайту https://www.python.org. На головній сторінці в секції Downloads наявно посилання на завантаження останньої версії (рис. А.1).

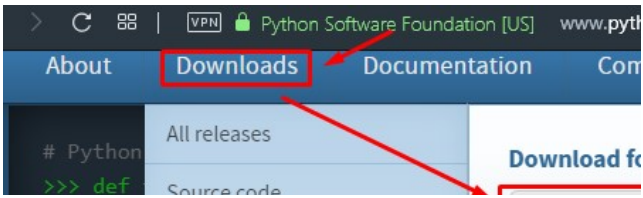

Рисунок А.1 –Завантаження інтерпретатору Python

Разом з інтерпретатором Python буде встановлено Pip - менеджер керування програмними пакетами, написаними на Python. Він знадобиться, якщо виникне необхідність встановлення додаткових програмних пакетів (бібліотек).

Для перевірки успішності встановлення програмного забезпечення слід відкрити командну строку і написати «python - h». При успішному встановленні інтерпретатору буде виведено інформацію про поточну версію програми.

За необхідністю встановлення додаткових програмних пакетів за допомогою Pip треба відкрити командну строку і ввести наступну команду «pip install package_name», де package_name змінити на ім'я пакету. Точну назву пакету можна перевірити на сайті-сховищі програмного забезпечення Python: https://pypi.org/.

Розробку програмного коду на Python можна виконувати навіть у блокноті, проте бажано використовувати середовище розробки(IDE). На відміну від стандартних текстових редакторів, IDE забезпечує повноцінну підтримку синтаксису, автодоповнення, можливість гарячого (швидкого) виконання створеного скрипту, а також багато іншого.

Одними з найпоширених і найпотужних середовищ розробки є:

- PyCharm (https://www.jetbrains.com/pycharm/download);

- Visual Studio (https://visualstudio.microsoft.com/downloads);
- Eclipse з розширенням PyDev (https://www.eclipse.org/downloads).

Змінні і їх особливості

Python є мовою з динамічною типізацією. Він визначає тип даних змінної виходячи із значення, яке їй присвоєно, а, отже, не потрібно явно вказувати тип змінних.

Для запобігання виникнення помилок при розробці програмного коду може виникнути необхідність дізнатись поточний тип змінної. Для цього слід викликати функцію type (variable_name), де variable_name – ім'я змінної.

Щоб перетворити змінну до іншого типу існують спеціальні функції: int(number), float(number). Для округлення змінної можна використати функцію round(number, count_decimal_places).

Арифметичні операції

Програмісту доступні такі арифметичні операції: +, -, *, /, // (цілочисельне ділення), ** (піднесення у ступінь: number ** power), % (залишок від ділення). Окрім цього наявні спеціальні оператори, які дозволяють присвоїти результат першому операнду: +=, -=, *=, /=, //=, **=, %=.

Введення і виведення через командний рядок, коментарі

Для виведення даних у командний рядок слід викликати команду «print(variable_name)». У випадку коли треба вивести декілька змінних їх можна вказувати через кому, наприклад «print(variable_name1, variable_name2)».

Для вводу даних у програму через командний рядок слід викликати команду «input()». Команда поверне введений рядок. Коли перед вводом треба вивести певне повідомлення – його можна вказати у душках, обгорнувши в лапки: «input("Message")». Наприклад, для зчитування варіанту студенту слід використати наступний код: v = int(input("Enter option: ")).

Для підвищення наочності коду і поліпшення його розуміння в коді можна залишати коментарі. На початку коментаря слід поставити символ решітки (#). Зверніть увагу, дія знаку розповсюджується лише на те, що стоїть після нього у тому ж самому рядку.

Деякі середовища розробки (наприклад PyCharm) підтримують виділення частин з певним коментарем. Для цього над ділянкою коду написати сполучення «# region» і в тому ж рядку коментар. Після коду, який «обгортають» слід написати наступне сполучення "# endregion".

Умовні конструкції

Умовні конструкції призначені для виконання різних алгоритмів в залежності від значень умовних виразів.

Одною з таких конструкцій є «if – else». Вона має наступне формальне визначення:

if логічний_вираз :
    інструкції
[elif логічне вираз :
    інструкції]
[else :
    інструкції]
(те, що у квадратних дужках не є обов'язковим)

Якщо логічний вираз повертає True, то виконується наступний після виразу блок коду. Весь блок повинен мати рівний відступ. Якщо в if повертає False то блок коду, який написано через відступ пропускається, перевіряється логічний вираз в elif (за наявності), якщо і він буде хибним – буде виконаний блок коду після else (за наявності).

В логічних виразах можуть бути застосовані наступні операції порівняння або їх комбінації: == (дорівнює), != (не дорівнює), > (більше),< (менше),>= (більше або дорівнює), <= (менше або дорівнює). Окрім цього, в логічних виразах можуть бути застосовані наступні логічні операції або їх комбінації: and (і), or (або), not (заперечення). Якщо виконується умова, яка побудована з логічних операцій і операцій порівняння виконується – логічний вираз повертає True, якщо не виконується - False.

Наприклад вирази:

$$S = \begin{cases} 10V, V < 10, \\ 5V, 10 \le V < 20, \\ 3V, V \ge 20; \end{cases} \qquad N = \begin{cases} 5V, V < 7, \\ 4V, 7 \le V < 10, \\ 3V, 10 \le V < 20, \\ 2V, V \ge 20; \end{cases}$$

можна запрограмувати таким чином:

```
s = v                n = v
if v < 10:           if  v < 7:
   s  *= 10             n *= 5
elif v < 20:         elif v < 10:
   s *= 5               n *= 4
else:                elif v < 20:
   s *= 3               n *= 3
                     else:
                        n *= 2
```

Допускається використання спрощеного формату умовної конструкції. У випадку, коли одній змінній слід призначити різні значення в залежності від певної умовної конструкції можливий такий запис: variable_name = value1 if conditional_expression else value2, де variable_name – ім'я змінної, якій буде призначено нове значення, value1 – значення, яке треба привласнити, якщо умова вірна, value2 – значення, яке треба привласнити, якщо умова не вірна, conditional_expression – логічний вираз.

Робота з масивами

При роботі з багатомірними масивами даних рекомендовано встановити пакет «NumPy». NumPy – це пакет наукових обчислень з Python, він містить: потужну функціональність для роботи з багатомірними масивами даних, функціонал лінійної алгебри та багато іншого.

Щоб встановити пакет у командному рядку треба викликати команду «pip install numpy». Після успішного встановлення пакету на комп'ютер його можна буде використовувати в програмах, на початку яких слід підключити пакет вказавши «import numpy». Для скорочення виклику можна призначити синонім, наприклад «import numpy as np».

Створення масиву проініціалізованого нулями можна зробити викликавши функцію «numpy.zeros(shape , dtype = type)», де shape - розмірність масиву (наприклад: (10, 5)), a type – тип значень (int або float).

Індекси елементів масивів починаються с нуля. Для перебору елементів масиву можна застосувати цикл for, наприклад: for i in range(size):, де i – буде індексом елементу, а замість size слід вказати розмір. Звертання до елементу одномірного масиву - «array_name[i]», у двомірному масиві це буде звертання до рядку, який відповідає індексу i.

Як приклад, вирази:

$$x_j^s = \begin{cases} jV - 0{,}1s,\ j = 1,5,9,..., \\ 0{,}01jV^{-1} + 0{,}3s,\ j = 2,4,6,..., \\ jrand,\ j = 3,7,11,...; \end{cases} \qquad y^s = \begin{cases} 0,\ (x_1^s)^2 + (x_2^s)^2 < V^2 + 0{,}04S^2, \\ 1,\ (x_1^2)^2 + (x_2^2)^2 \geq V^2 + 0{,}04S^2; \end{cases}$$

можна запрограмувати наступним чином:

```
import numpy
import random
…
x = np.zeros((s, n), dtype=np.float)
for i in range(s):
  for j in range(0, n, 4):
    x[i][j] = (j + 1) * v - 0.1 * (i + 1)
  for j in range(1, n, 2):
    x[i][j] = 0.01 * (j + 1) / v + 0.3 * (i
```

```
import numpy
…
y = np.zeros((s, 1), dtype=np.int)
for i in range(s):
  if pow(x[i][0], 2) + pow(x[i][1], 2) >=
pow(v, 2) + 0.04 * pow(v, 2):
    y[i] = 1
```

```
+ 1)
  for j in range(2, n, 4):
    x[i][j] = random.random()
```

В цьому прикладі також було використати генерацію випадкових чисел - вбудовану в пакет random функцію random() – яка генерує число з від 0 до 1. Пакет random входе до базового набору пакетів, тож його не потрібно встановлювати, проте слід імпортувати до програми «import random».

При виконанні лабораторних робіт можуть знадобитись такі функції:

- отримання масиву мінімумів з стовпчиків чи строк двомірного масиву - «np.amin(array_name, axis=value)», де array_name – ім'я двомірного масиву, а value – значення 0, якщо потрібно отримання в стовпчиків, 1 – з строк;
- отримання масиву максимумів з стовпчиків чи строк двомірного масиву - «np.amax(array_name, axis=value)», аргументи такі ж як у «np.amin»;
- отримання масиву середніх значень з стовпчиків чи строк двомірного масиву - «np. mean (array_name,axis=value)», аргументи такі ж як у «np.amin»;
- отримання масиву, кожен елемент якого піднесено у ступінь – «pow(array_name, power)», де power - ступінь;
- отримати суму елементів масиву - sum(array_name);
- змінити формат масиву – «array_name.reshape(count_el_height, count_el_width)» – де count_el_height – кількість строк, а count_el_width – кількість стовпчиків, якщо параметр треба розрахувати автоматично – слід вказати «-1».

Функції

Функції - це блок коду, який виконує певне завдання і який можна повторно використовувати. Розподіл коду на функції сприяє полегшенню сприйняття програми, використання функції вважається гарною практикою.

Формальне визначення функції:

```
def func_name ([options]):
    code...
    [return value]
```

, де def – ім'я яким позначаються функції, func_name – ім'я функції, options – змінні, які передаються у функцію (за наявності), code... – інструкції, які виконує функція, return value значення, яке повертає функція (за наявності). Тіло функції потрібно писати з однаковим відступом.

Наприклад, вищенаведений код з розділу «Робота з масивами» можна оформити у функцію:

```
import numpy
…
def get_y(s, v, x):
  y = np.zeros((s, 1), dtype=np.int)
```

```
  for i in range(s):
    if pow(x[i][0], 2) + pow(x[i][1], 2) >= pow(v, 2) + 0.04 * pow(v, 2):
      y[i] = 1
  return y
```

Функцію можна буде викликати наступним чином:

```
y = get_y(s, v, x)
```

Час виконання

Одним з варіантів отримання часу виконання певної частини коду є функція «default_timer» з пакету «timeit». Цей варіант може вимірювати навіть величини, які менші за 0.1 секунди. Для отримання часу виконання частини коду слід зберігти значення функції «default_timer()» перед виконанням ділянки коду, а після виконання ділянки коду ще раз викликати функцію «default_timer()» і відняти від неї раніше збережене значення.

Наприклад, що б з'ясувати час виконання функції get_y(s, v, x) треба:

```
from timeit import default_timer
…
start_time = default_timer ()
y = get_y(s, v, x)
time = default_timer () - start_time
```

Поради до виконання елементів лабораторних робіт

Завантажити файл вибірки для аналізу можна за посиланням: https://www.kaggle.com/c/titanic/data (рис. А.2).

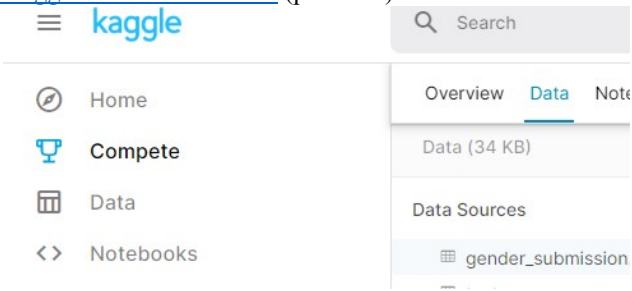

Рисунок А.2 – Завантаження вибірки train.csv

Для завантаження даних у програму з файлу формату «csv» слід застосувати функцію «read_csv(path)», з пакету «pandas», path – шлях до файлу, якшо файл вибірки знаходиться в теці з файлом коду достатньо вказати лише назву файла, наприклад: «data_name = pandas.read_csv('train.csv')».

Вивести верхівку вибірки можна командою «print(data_name.head())».

Видалення певних ознак реалізується командою «data_name.drop([NameCol_1, …, NameCol_N], axis=1)», де NameCol – ім'я ознаки. При виборі ознак для видалення з вибірки «Titanic train» особливу увагу радимо звернути на «PassengerId», «Name», «Ticket», «Cabin».

Алгоритми класифікації з «Sklearn» потребують, що б у вибірки не було відсутніх значень, тобто слід перевірити перед класифікацією вибірку на відсутні значення вибірку. Переглянути кількість відсутніх значень можна функцією «print(data_name.isnull().sum())».

При виявленні відсутніх значень – слід їх заповнити. Якщо всі пропуски певної ознаки заповнюють однім значенням – використовують функцію «data_name.sign_name.fillna(value, inplace=True)» (наприклад: заповнення ознаки «вік» медіаною з значень).

Алгоритми класифікації з пакету «Sklearn» не можуть роботи з символьними даними. Їх потрібно закодувати. Наприклад: ознаку «стать людини» можна закодувати: «male» - це 1, а «female» - це 0 (чи навпаки).

Перед проведенням аналізу слід відділити цільову ознаку від інших. Це можна зробити наступним чином: «data_target = data_name.sign_name». Не збудьте видалили цільову ознаку з інших: «data_train = data_name.drop([sign_name], axis=1)».

Ініціалізацію алгоритмів можна використовувати з параметрами за замовчуванням, наприклад: «model_knc=KNeighborsClassifier()», або з вказанням бажаних параметрів: «model_knc=KNeighborsClassifier(n_neighbors=18)».

Перелік потрібних алгоритмів і модулів, які для них необхідні:

- KNeighborsClassifier - sklearn.neighbors;
- RandomForestClassifier - sklearn.ensemble
- LogisticRegression - sklearn.linear_model
- svm - sklearn

Один з способів проведення класифікації і тестування є застосування так званого «ковзаючого контролю» - «cross_val_score», наприклад: «cross_val_score(model_name, data_train, data_target)». Ця функція поверне набір значень, отриманих під час декількох тестувань. Для оцінювання результативності алгоритму можна використати усереднене значення з них.

Побудову графіку можна зробити за допомогою функціоналу «pyplot» з пакету «matplotlib». Спочатку слід проініціалізувати фігуру (графік), потім задати значення, підписати стовпці і вивести побудований графік. Враховуючи, що будування графіку присутнє і в наступних лаб. роботах рекомендовано код побудови відокремити у функцію.

Приклад побудови графіку, де «result» - словник з даними «назва стовпця – значення»:

```
plt.figure()  # Ініціалізація нового графіку
plt.title(title)  # Наіменування графіку
```

plt.bar(range(len(data)), data.values(), color=color) # Встановлення параметрів стержневої діаграми
plt.xticks(range(len(data)), data.keys()) # Встановлення підпису стовпців діаграми
plt.show() # Друкування графіку

Для завантаження даних у програму для відбору ознак слід використати функцію «dataset_name = datasets.load_iris()», попередньо підключивши відповідний модуль до програми(«from sklearn import datasets»). Вивести перелік ознак вибірки можна командою «print(dataset_name.feature_names)».

Для полегшення сприйняття значень коефіцієнтів можна обмежити кількість знаків після коми при друкуванні масивів командою: «numpy.set_printoptions(precision=number)», де number – бажана кількість знаків.

Данні за ознаками можна отримати викликавши «dataset_name.data», значення розподілу можна отримати викликавши «dataset_name.target». Для зручного користування їх можна привласнити змінним з коротшими назвами, наприклад: x, y.

Ініціалізацію алгоритмів можна використовувати з параметрами за замовчуванням, наприклад: «skb = SelectKBest()», або з вказанням бажаних параметрів: «skb = SelectKBest(chi2, k=2)». Для розрахування параметрів слід застосувати функцію «fit», наприклад «fit = skb.fit(x, y)», після чого слід застосувати перетворення за розрахованими параметрами до певного набору даних, наприклад: «data_sub = skb.transform(x)». Для послідовного виконання обох процесів є функція «fit_transform».

Нижче наведено перелік алгоритмів і методів, які необхідні, для виведення розрахованих коефіцієнтів (усі алгоритми імпортовані з модулю «feature_selection», окрім ExtraTreesClassifier – він з «sklearn.ensemble»):

- VarianceThreshold - variances_ (наприклад: vt = VarianceThreshold(), x_sub = vt.fit_transform(x), print(vt.variances_));

- SelectKBest - scores_ (наприклад: skb = SelectKBest(), fit = skb.fit(x, y), print(fit.scores_));

- RFE - estimator_.coef_ (наприклад: rfe = RFE(LogisticRegression()), rfe_est_coef = np.amax(rfe.fit(x[range(0, len(x), 2)], y[range(0, len(y), 2)]).estimator_.coef_, axis=0, print(rfe_est_coef));

- SelectFromModel - estimator_.coef_ (наприклад: sfm =SelectFromModel (estimator=LinearSVC().fit(x, y),prefit=True),print(np.amax(sfm.estimator.coef_, axis=0)));

- ExtraTreesClassifier - feature_importances_ (наприклад: etc = ExtraTreesClassifier(), fit = etc.fit(x, y), fit.feature_importances_).

*комами вище розділені рядки коду

Для отримання індексів відібраних ознак можна викликати функцію «get_support(indices=True)», наприклад: vt.get_support(indices=True).

Побудову графіку можна зробити за допомогою функціоналу «pyplot» з пакету «matplotlib», проте принцип побудови дещо відрізняється від того, що був раніше. Спочатку слід сформувати значення, після чого бажано підписати координатні вісі, проініціалізувати даними і вивести побудований графік.

Приклад побудови графіку:

```
x_axis, y_axis = dataset.data[:, index_1], dataset.data[:, index_2]  # Формування набору значень відповідно до обраних ознак, index – номер ознаки
plt.xlabel(dataset.feature_names[index_1])  # Підпис осі x
plt.ylabel(dataset.feature_names[index_2])  # Підпис осі y
plt.scatter(x_axis, y_axis, c=dataset.target)  # Ініціалізація графіку значеннями, з наданням точкам різного кольору відповідно до вихідного класу
plt.show()  # Виведення графіку
```

Для отримання усіх можливих комбінацій (без врахування порядку) можна використати функцію «combinations(range(value)), size)», де value – кількість елементів, а size – кількість елементів в комбінації, наприклад: «combinations(range(len(dataset_name.feature_names)),2)».

Перелік потрібних алгоритмів і модулів, які необхідні для алгоритмів:

- KMeans- sklearn. cluster;
- dendrogram - scipy.cluster.hierarchy.

Ініціалізацію алгоритму KMeans можна використовувати з параметрами за замовчуванням, наприклад: «model_KMeans = cluster.KMeans()», або з вказанням бажаних параметрів: «model_KMeans = cluster.KMeans(n_clusters=3)». Після ініціалізації слід застосувати функції «fit» і «predict».

Для отримання результатів розподілу слід використати функцію «pd.crosstab(target1, target2)», де target1 – це істинні значення цільової ознаки, а target2 – результати кластеризації.

При реалізації ієрархічної кластеризації слід спочатку згенерувати матрицю зв’язку за допомогою «linkage», наприклад: matrix = linkage(dataset_name.data, 'ward'). Після чого можна побудувати графік за яким визначити межі кластерів потрібного розміру (враховуючи особливості ієрархічної кластеризації). Для цього слід проініціалізувати фігуру («plt.figure(figsize=(20, 10))»), побудувати ієрархічну кластеризацію у вигляді дендрограми («hierar_cl = dendrogram(matrix, leaf_font_size=8)») і вивести графік («plt.show()»).

Після виведення графіку оператору слід ввести значення границь (лівої вистачить) 1го і 2го кластерів (ліва границя нульового – 0). Після цього віднести значення відповідно до кластерів, для чого спочатку створити порожню

матрицю (predictions_HC = np.zeros(len(hierar_cl), dtype=np.int)) і заповнити відповідними до результатів кластеризації значеннями.

Наприклад, розподіл може бути реалізовано наступним чином:

```
cluster_number = 1
for i in range(len(dataset_name.data)):
   if cluster_number == 1 and hierar_cl['leaves'][i] == left_border_cl_2:
      cluster_number = 2
   elif hierar_cl['leaves'][i] == left_border_cl_3:
      break
```

Після розподілу слід проаналізувати результати аналогічно до того, як зроблено з KMeans.

У разі необхідності провести дослідження вибірки з частковим вибором ознак – одразу після завантаження вибірки слід видалити з «dataset_name.data» і «dataset_name. feature_names» стовпчики, які заважають.