math_2022_038

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Факультет математики та інформатики
кафедра математичного моделювання

ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ У БІОСТАТИСТИЦІ

Кваліфікаційна робота

Рівень вищої освіти –другий (магістерський)

Виконав:
студент 6 курсу, 607 групи
Діков Максим Валентинович
Керівник:
кандидатфізико-математичних наук,
доцент Дорошенко І. В.

До захисту допущено
на засіданні кафедри
протокол № 8 від 6 грудня 2022 р.
Зав. кафедрою _____ проф. Черевко І.М.

Чернівці – 2022

Анотація

В даній магістерській роботі розглядаються статистичні підходи, які застосовуються у біоінформатиці. В роботі досліджується модельний приклад, де встановлюється зв’язок між фізичними характеристиками пацієнтів, шкідливими звичками, віком тощо та випадками інсульту. Аналіз зроблено за допомогою середовища R.

Ключові слова: біостатистика, машинне навчання, лінійна регресія, логістична регресія.

Annotation

This master's thesis examines statistical approaches used in bioinformatics. The work examines a model example, where the connection between physical characteristics of patients, bad habits, age, etc., and stroke cases is established. The analysis was done using the R environment.

Keywords: biostatistics, machine learning, linear regression, logistic regression.

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів наукових досліджень інших авторів мають посилання на відповідне джерело.

_____ М. В. Діков
(підпис)

Зміст

Вступ

Методи машинного навчання все частіше використовуються для вирішення проблем біоінформатики. Нові обчислювальні методи аналізу високопродуктивних даних у формі послідовностей, експресій генів і білків, шляхів і зображень стають життєво важливими для розуміння хвороб і виродництва нових ліків. Методи машинного навчання, такі як моделі Маркова, опорні векторні машини, нейронні мережі та графічні моделі, успішно зарекомендували себе в аналізі наукових даних про життя завдяки їхнім можливостям обробки випадкового та невизначеного шуму даних із подальшим його узагальненням. Міжнародно визнані групи видатних дослідників у галузі машинного навчання в біоінформатиційній сфері - користуються методами машинного навчання для вирішення сучасних проблем у біоінформатиці. Таким чином можна проводити наступні обчислення:

- підбірознак аналізу геномних і протеомних даних;
- порівняння методів відбору змінних у відборі генів та класифікації даних мікрочипів;
- нечіткий генний аналіз;
- прогнозування на основі послідовності властивостей на рівні білкових залишків;
- відбір найімовірнішихсистем довгострокових ознак у біопослідовностях; і набагато більше.

Машинне навчання в біоінформатиці є незамінним ресурсом для інформатиків, інженерів, біологів, математиків, дослідників, клініцистів та лікарів.

Розділ 1. Машинне навчання в біоінформатиці

1.1. Загальні відомості

Експоненціальне зростання кількості доступних біологічних даних породжує такі аспекти: з одного боку, ефективне зберігання інформації та управління нею, а з іншого боку, складність вилучення корисної інформації з цих даних. Другий аспект є однією з головних проблем у комп'ютерній біології, який вимагає розробки інструментів і методів, здатних перетворити всі ці різнорідні дані в корисні біологічні знання. Ці інструменти та методи повинні дозволити нам вийти за рамки простого опису даних і надати знання у формі тестованих моделей. За допомогою цієї спрощеної абстракції, ми можемо будувати системи для прогнозування.

Існує кілька біологічних областей, де методи машинного навчання застосовуються для отримання корисних знань із даних. На рисунку нижче представлена схема основних біологічних задач, у яких застосовуються обчислювальні методи. Тут ці проюблеми поділені на шість різних областей: "геноміка", "протеоміка", "мікрочипи", "системна біологія", "еволюція" та "пошук тексту"[1]. Категорія під назвою "інше" об'єднує проблеми, що залишилися. Ці категорії слід розуміти дуже загально, особливо геноміку та протеоміку, які в цьому огляді розглядаються як дослідження нуклеотидних ланцюгів і білків відповідно.

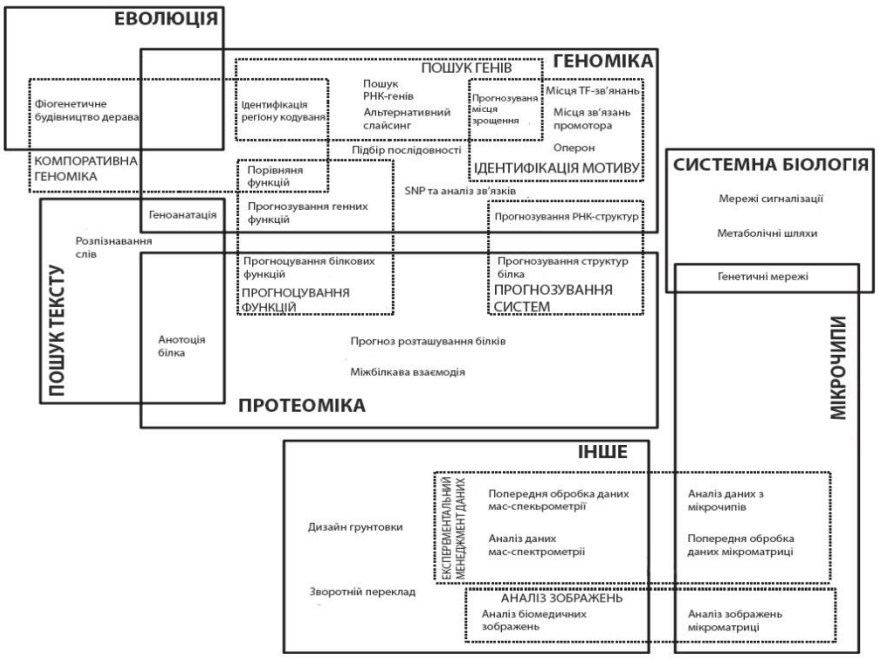


Рис.1.1. Схема основних біологічних задач

Геноміка є однією з найважливіших областей біоінформатики. Кількість доступних послідовностей зростає експоненціально, як показано на малюнку нижче. Ці дані потрібно обробити, щоб отримати корисну інформацію. Як перший крок, з послідовностей генома ми можемо виділити

розташування та структуру генів. Останнім часом ідентифікація регуляторних елементі і некодуючих генів РНК також розглядається з обчислювальної точки зору. Інформація про послідовність також використовується для прогнозування функції гена та вторинної структури РНК.

Якщо гени містять інформацію, білки є працівниками, які цю інформацію використовують. Білки відіграють дуже важливу роль у життєвому процесі, а їх тривимірна (3D) структура є ключовою особливістю їх функціональності. У протеомній області основним застосуванням обчислювальних методів є передбачення структури білка. Білки - це дуже складні макромолекули з тисячами атомів і меж. Отже, кількість можливих структур величезна. Це робить прогнозування структури білка дуже складною комбінаторною проблемою, де потрібні методи оптимізації. У протеоміці, як і у випадку геноміки, методи машинного навчання застосовуються для прогнозування функції білка.

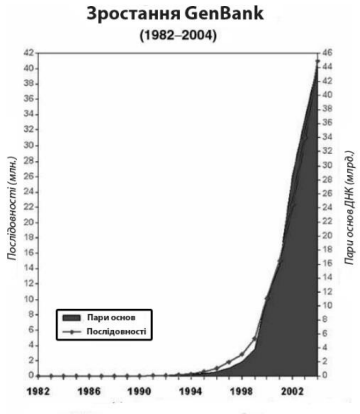


Рис. 1.2.

Іншим цікавим застосуванням обчислювальних методів у біології є керування складними експериментальними даними. Випробування мікрочипів є найвідомішою (але не єдиною) областю, де збираються такі дані. Складні експериментальні дані породжують дві різні проблеми. По-перше, дані мають бути попередньо оброблені, тобто модифіковані для належного використання алгоритмами машинного навчання. По-друге, дані треба проанолізувати в залежиності від того, що ми шукаємо. У випадку мікроматричних даних найбільш типовими застосуваннями є ідентифікація шаблонів експресії, класифікація та індукція генетичної мережі.

Системна біологія — ще одна область, де біологія та машинне навчання працюють разом. Дуже складно моделювати життєві процеси, що відбуваються всередині клітини. Таким чином, обчислювальні методи надзвичайно корисні при моделюванні біологічних мереж, особливо генетичних мереж, мереж передачі сигналу та метаболічних шляхів.

Еволюція і, особливо, реконструкція філогенетичного дерева також використовують переваги методів машинного навчання. Філогенетичні дерева — це схематичні зображення еволюції організмів. Традиційно вони конструювалися відповідно до різних ознак (морфологічних особливостей, метаболічних особливостей тощо), але в наш час, з великою кількістю доступних послідовностей геномів, алгоритми побудови філогенетичного дерева базуються на порівнянні між різними геномами. Це порівняння здійснюється за допомогою множинного вирівнювання послідовностей, де дуже зручно використовивати методи оптимізації.

Побічним ефектом застосування обчислювальних методів до зростаючої кількості даних є збільшення доступних публікацій. Це забезпечує нове джерело цінної інформації, де для вилучення знань потрібні методи інтелектуального аналізу тексту. Таким чином, інтелектуальний аналіз тексту стає все більш зничимим в обчислювальній біології, і його застосовують у функціональній анотації, прогнозуванні клітинного розташування та аналізі взаємодії білків.

На додаток до всього вище перечисленого обчислювальні методи використовуються для вирішення інших проблем, таких як ефективний дизайн праймерів для ПЛР, аналіз біологічних зображень і зворотна трансляція білків (що, враховуючи виродження генетичного коду, є складною комбінаторною задачею).

Машинне навчання потрібно для оптимізації критерію продуктивності за допомогою прикладів даних або минулого досвіду. Оптимізованим критерієм може бути точність, що забезпечується моделлю прогнозування - в задачі моделювання, і значення функції придатності чи оцінки - в задачі оптимізації

В задачі моделювання термін «навчання» означає запуск комп’ютерної програми для створення моделі за допомогою навчальних даних або минулого досвіду. Машинне навчання використовує статистичну теорію під час створення обчислювальних моделей, оскільки мета полягає в тому, щоб зробити висновки на основі вибірки. Двома основними кроками в цьому процесі є створення моделі шляхом обробки величезної кількості даних і представлення моделі та ефективне створення висновків. Необхідно зауважити, що ефективність алгоритмів навчання та логічного висновку, а також їх просторова та часова складність, прозорість та можливість інтерпретації можуть бути такими ж важливими, як і точність прогнозування. Процес перетворення даних у знання є як ітеративним, так і інтерактивним. Ітераційна фаза складається з кількох кроків. На першому кроці нам потрібно інтегрувати та об’єднати різні джерела інформації в один формат. На другому кроці необхідно вибрати, очистити та трансформувати дані. Щоб

виконати цей крок, нам потрібно виключити або виправити невиправлені дані, а також визначити стратегію імпутації відсутніх даних. На цьому кроці також вибираються релевантні та незайві змінні; цей вибір також можна зробити щодо екземплярів. На третьому етапі, який називається інтелектуальним аналізом даних, ми беремо до уваги цілі дослідження, щоб вибрати найбільш відповідний аналіз даних. На цьому кроці слід вибрати тип парадигми для контрольованої або неконтрольованої класифікації, і модель буде створена з даних. Коли модель отримана, її слід оцінити та інтерпретувати - як зі статистичної, так і з біологічної точок зору - і, якщо необхідно, повернутися до попередніх кроків для нової ітерації. Це включає вирішення конфліктів з поточними знаннями в області. Задовільно перевірена модель - і виявлені нові знання — потім використовуються для вирішення проблеми.

Проблеми оптимізації можна поставити як задачу пошуку оптимального рішення в просторі кількох (іноді експоненціального розміру) можливих рішень. Вирішальним є вибір методу оптимізації, який буде використано для вирішення проблеми. Оптимізаційні підходи до біологічних проблем можна класифікувати, залежно від типу знайдених рішень, на точні та наближені методи. Точні методи виводять оптимальні рішення, коли досягається збіжність. Однак вони не обов'язково збігаються для кожного випадку. Наближені алгоритми завжди виводять варіант рішення, але не гарантовано те, що він буде оптимальним.

Оптимізація також є фундаментальним завданням при моделюванні на основі даних. Насправді процес навчання на основі даних можна розглядати як пошук моделі, яка найкраще підходить для даних. У цьому пошуку в просторі моделей може бути використаний будь-який тип евристики. Таким чином, методи оптимізації також можна розглядати як складову в моделюванні.

1.2. Контрольована класифікація

У задачі класифікації ми маємо набір елементів, розділених на класи. Даному елементу (або екземпляру) множини - клас призначається відповідно ознак елемента та набору правил класифікації. У багатьох ситуаціях реального життя цей набір правил невідомий, і єдиною доступною інформацією є набір позначених прикладів (тобто набір екземплярів, пов’язаних із класом). Контрольовані парадигми класифікації — це алгоритми, які індукують правила класифікації з даних.

Як приклад, ми побачимо можливий спосіб вирішення проблеми прогнозування місця зрощування як контрольованої проблеми класифікації. Екземпляри, що підлягають класифікації, будуть послідовностями ДНК

заданого розміру. Атрибутами даного екземпляра буде нуклеотид у кожній позиції в послідовності. У прикладі ми припустимо, що шукаємо донорські сайти, тому можливими значеннями для класу будуть справжній донорський сайт або помилковий донорський сайт. Оскільки ми підходимо до проблеми як контрольованої класифікації, нам потрібен набір прикладів із мітками, тобто набір послідовностей справжніх і хибних донорських сайтів разом із їхньою міткою. На цьому етапі ми можемо використовувати цей навчальний набір для створення класифікатора. Після навчання класифікатора ми можемо використовувати його для позначення нових послідовностей, використовуючи нуклеотид, присутній у кожній позиції, як вхідні дані для класифікатора та отримуючи призначену мітку (справжній або хибний донорський сайт) як вихідні дані.

У двогруповій контрольованій класифікації існує вектор ознак $X \in \mathbb{R}^n$, компоненти якого називаються змінними предикторами та змінна мітки або класу $C \in \{0, 1\}$. Отже, завдання полягає в тому, щоб індукувати класифікатори з навчальних даних, які складаються з набору N незалежних спостережень $D_n = \{(x^{(1)}, c^{(1)}), \ldots, (x^{(N)}, c^{(N)})\}$, отриманого із спільного розподілу ймовірностей $p(x,c)$, як показано в таблиці нижче.

Таблиця 1.1. Необроблені дані в задачі контрольованої класифікації

	X_1	...	X_n	C
$(\boldsymbol{x}^{(1)}; c^{(1)})$	$x_1^{(1)}$	...	$x_n^{(1)}$	$c^{(1)}$
$(\boldsymbol{x}^{(2)}; c^{(2)})$	$x_1^{(2)}$	...	$x_n^{(2)}$	$c^{(2)}$
$(\boldsymbol{x}^{(N)}; c^{(N)})$	$x_1^{(N)}$	...	$x_n^{(N)}$	$c^{(N)}$
$\boldsymbol{x}^{(N+1)}$	$x_1^{(N+1)}$	...	$x_n^{(N+1)}$	???

Модель класифікації буде використовуватися для призначення міток новим екземплярам відповідно до значення її змінних предикторів.

1.3. Оцінка та порівняння алгоритмів класифікації. Частота помилок і крива ROC

Коли використовується втрата , усі помилки однаково погані, і наші розрахунки помилок базуються на матриці невідповідностей (табл. 1.2).

Таблиця 1.2.

		Прогнозований клас	
		Позитивний	**Негативний**
Істинний клас	Позитивний	TP: Істинно позитивний (true positive)	FN: Ложно негативний (false negative)
	Негативний	FP: Ложно позитивний (false positive)	TN: Істинно негативний (true negative)

У цьому випадку ми можемо визначити частоту помилок як , де $N = |TP| + |FP| + |TN| + |FN|$ це загальна кількість екземплярів у наборі перевірки.

Для точного налаштування класифікатора існує інший підхід, який полягає в тому, щоб намалювати криву робочих характеристик приймача (ROCs), яка показує частоту попадань проти частоти помилкових тривог, а саме, проти , і має форму, подібну до малюнку нижче:

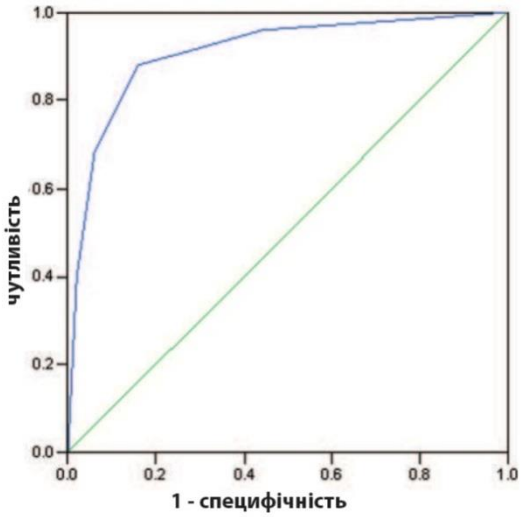


Рис.1.3. Приклад кривої ROC

Для кожного алгоритму класифікації існує параметр, наприклад, поріг прийняття рішення, зякий ми можемо змынювати, щоб змінити кількість справжніх позитивних результатів проти помилкових позитивних. Збільшення кількості справжніх спрацьовувань також збільшує кількість помилкових тривог; зменшення кількості помилкових тривог також зменшує кількість звернень. Залежно від того, наскільки вони хороші/важливі для конкретного застосування, ми вибираємо точку на цій кривій. Площа під кривою робочої характеристики приймача використовується як міра продуктивності для алгоритмів машинного навчання.

Оцінка похибки класифікації

Важливим питанням, пов’язаним із розробленим класифікатором, є те, як оцінити його (очікувану) частоту помилок під час використання цієї моделі для класифікації невидимих (нових) екземплярів.

Найпростіший і найшвидший спосіб оцінити похибку розробленого класифікатора за відсутності тестових даних — це обчислити його похибку на самих вибіркових даних. Ця оцінка повторної підстановки дуже швидка для обчислення і зазвичай є оптимістичною (тобто з низьким зміщенням) оцінкою справжньої помилки.

У k-кратній перехресній перевірці D_N розбивається на k згорток. Кожена згортка не входить у процес проектування та використовується як набір для тестування. Оцінка помилки - це загальна частка помилок, допущених під час усіх згорток. Під час перехресної перевірки з одним виходом одне спостереження щоразу пропускається, що відповідає N-кратній перехресній перевірці.

Методологія початкового завантаження — це загальна стратегія передискретизації, яка може бути застосована для оцінки помилок. Віно заснована на понятті «емпіричного розподілу», який надає масу для кожної з N точок даних. «Початкова вибірка», отримана з цього «емпіричного розподілу», складається з N рівноімовірних малюнків із заміною з вихідного набору даних D_N. Використовується початкова нульова оцінка та початкова оцінка 0,632.

Вибір підмножини функцій

Одне питання, яке є спільним для всіх контрольованих парадигм класифікації, полягає в тому, чи всі n ознак опису корисні під час вивчення правила класифікації. При спробі відповісти на це запитання виникає так звана проблема вибору підмножини ознак (FSS), яку можна переформулювати так: за наявності набору ознак-кандидатів виберіть найкращу підмножину за певним алгоритмом навчання.

Це зменшення розмірності, виконане процесом FSS, може принести кілька переваг контрольованій системі класифікації, наприклад, зниження вартості збору даних, покращення розуміння кінцевої моделі класифікації, швидке впровадження остаточної моделі класифікації та підвищення точності класифікатора.

FSS можна розглядати як задачу пошуку, де кожен стан у просторі пошуку визначає підмножину можливих характеристик завдання. Вичерпна оцінка можливих підмножин ознак зазвичай неможлива на практиці через велику кількість обчислювальних зусиль. Чотири основні питання визначають природу процесу пошуку: початкова точка простору пошуку,

організація пошуку, функція оцінки підмножини ознак і критерій зупинки пошуку.

Початкова точка простору пошуку визначає напрямок пошуку. Можна почати без функцій і послідовно додавати їх, або можна почати з усіх функцій і послідовно їх видаляти. Можна також вибрати початковий стан десь посередині простору пошуку.

Організація пошуку визначає стратегію пошуку в просторі розміром 2^n, де n – кількість ознак у задачі. Стратегії пошуку можуть бути оптимальними або евристичними. Існує два класичних оптимальних алгоритму пошуку, які вичерпно оцінюють усі можливі підмножини, - це depth-first і breadth-first. В іншому випадку пошук за розгалуженнями та межами гарантує виявлення оптимальної підмножини для монотонних оціночних функцій без систематичного дослідження всіх підмножин. Коли монотонність не може бути задовільнена, залежно від кількості ознак і використовуваної функції оцінки, вичерпуючий пошук може бути непрактичним. У цій ситуації евристичний пошук цікавий, оскільки він може знаходити майже оптимальні рішення, якщо не абсолютно оптимальні. Серед евристичних методів розрізняють детермінований і стохастичний алгоритми. З одного боку, класичні детерміновані евристичні алгоритми FSS — це послідовний прямий і зворотний вибір, плаваючі методи вибору або найкращий пошук. Вони є детермінованими в тому сенсі, що всі цикли завжди отримують одне й те саме рішення, і через свою природу, що піднімається на гору, вони мають тенденцію потрапляти в пастку на локальних піках, викликаних взаємозалежністю між функціями. З іншого боку, стохастичні евристичні алгоритми FSS використовують випадковість, щоб уникнути локальних максимумів, що означає, що не слід очікувати того самого рішення від різних прогонів. Генетичні алгоритми та алгоритми оцінки розподілу застосовують до задач FSS.

Функція оцінки вимірює ефективність певної підмножини функцій після того, як алгоритм пошуку обрав її для перевірки. Кожна підмножина ознак, запропонована алгоритмом пошуку, оцінюється за допомогою критерію (точність, площа під кривою ROC, взаємна інформація щодо змінної класу тощо), який слід оптимізувати під час пошуку. У так званому підході обгортки до проблеми FSS, алгоритм проводить пошук хорошої підмножини ознак, використовуючи помилку, яку повідомляє класифікатор як критерій оцінки підмножини ознак. Однак, якщо алгоритм навчання не використовується у функції оцінювання, якість підмножини ознак можна оцінити лише з огляду на внутрішні властивості даних. Алгоритм навчання з’являється лише в заключній частині процесу FSS для побудови остаточного класифікатора з використанням набору вибраних ознак. У статистичній

літературі пропонується багато заходів для оцінки якості підмножини ознак-кандидатів. Такий підхід до FSS називається фільтром у сфері машинного навчання.

Що стосується критерію зупинки пошуку, інтуїтивним підходом є погіршення значення функції оцінювання альтернативних підмножин. Іншим класичним критерієм є фіксація ряду можливих рішень, які потрібно відвідати під час пошуку.

Застосування методології FSS до даних мікрочіпів намагається отримати надійну ідентифікацію диференціально експресованих генів. Найбільш звичайним підходом до FSS у цій області є підхід фільтра, через величезну кількість функцій, з яких ми отримуємо інформацію.

1.4. Класифікаційні парадигми

У цьому підрозділі буде представлено основні характеристики деяких найбільш репрезентативних парадигм класифікації. Слід зазначити, що в такій області, як біоінформатика, де відкриття нових знань має велике значення, слід також враховувати прозорість та інтерпретацію парадигми.

Кожна контрольована парадигма класифікації має асоційовану поверхню прийняття рішень, яка визначає тип проблем, які класифікатор може вирішити. Немає єдиного найкращого класифікатора для всіх можливих навчальних наборів.

Байєсівські класифікатори мінімізують загальну вартість неправильної класифікації, використовуючи таке призначення: $y(x) = \arg min_k \sum_{c=1}^{r0} cost(k,c)p(c|x_1, x_2, \dots, x_n)$, де *cost(k, c)* позначає вартість неправильної класифікації. У випадку функції втрат 0/1 байєсівський класифікатор призначає найбільш ймовірний апостериорний клас даному екземпляру, тобто: $y(x) = \arg min_c\, p(c|x_1, x_2, \dots, x_n) = \arg min_c\, p(c)p(x_1, x_2, \dots, x_n|c)$. Залежно від способу апроксимації $(x_1, x_2, \dots, x_n|c)$ отримують байєсовські класифікатори різної складності.

Найпростішим байєсівським класифікатором є *Naive Bayes*. Він побудований на основі припущення про умовну незалежність прогностичних змінних, заданих класом. Хоча це припущення неодноразово порушується в реальних областях, парадигма все ще добре працює в багатьох ситуаціях. Найімовірніше апостериорне присвоєння змінної класу обчислюється як

$$c^* = arg\ \max_c p(c|x_1, \dots, x_n) = arg\ \max_c p(c) \prod_{i=1}^{n} p(x_i|c).$$

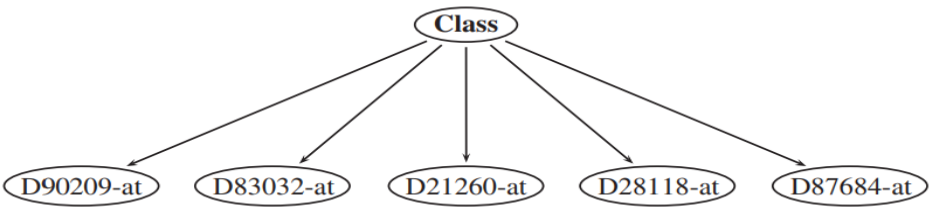


Рис.1.4. Класифікатор Байєса

Напівнадійний класифікатор Байєса намагається уникнути припущень про умовну незалежність прогностичних змінних, враховуючи змінну класу та нові змінні. Ці нові змінні складаються зі значень декартового добутку змінних області, що долають умову. Умова пов'язана з концепцією незалежності та надійністю оцінок умовної ймовірності.

Наївний класифікатор Байєса з доповненим деревом також враховує зв'язки між прогнозними змінними шляхом розширення наївної байєсівської структури за допомогою деревовидної структури серед прогнозних змінних. Ця деревовидна структура отримана шляхом адаптації алгоритму, запропонованого Чоу та Ліу, і обчислення умовної взаємної інформації для кожної пари прогнозних змінних, заданого класу. Доповнена деревом наївна модель класифікації Байєса обмежена кількістю батьків прогнозних змінних. У ньому прогностична змінна може мати максимум двох батьків: клас та іншу прогностичну змінну. Байєсівський класифікатор залежності k (kDB) уникає цього обмеження, дозволяючи передбачуваній змінній мати до k батьків, окрім класу.

Парадигма логістичної регресії визначається як ; де *x* представляє примірник, який потрібно класифікувати, а *0, 1, ..., n* є параметрами моделі. Ці параметри слід оцінити з даних, щоб отримати конкретну модель. Оцінка параметра здійснюється за допомогою методу оцінки максимальної правдоподібності. Система з *n+1* рівнянь і *n+1* параметрів, які потрібно розв'язати, не має аналітичного розв'язку. Таким чином, оцінки максимальної правдоподібності отримують ітераційним способом. Процедура Ньютона-Рафсона в цьому випадку є стандартною.

Процес моделювання базується на тесті Вальда та на тесті відношення правдоподібності. Пошук у просторі моделей зазвичай здійснюється за допомогою прямого, зворотного або покрокового підходів.

*Дискримінантний аналіз.*Лінійний дискримінантний аналіз Фішера заснований на знаходженні лінійних комбінацій, *xw*, *n*-вимірних значень змінної предиктора $x = (x_1, \dots, x_n)$, з великими співвідношеннями сум квадратів між групами та всередині групи. Для $N \times (n + 1)$ матриці набору даних співвідношення сум квадратів міжгрупових і внутрішньогрупових задається

w'Bw/w'Ww, де B і W позначають n×n матриць міжгрупових і внутрішньогрупових сум квадратів і перехресних добутків. Екстремальні значення*w'Bw/w'Ww* отримані з власних значень і власних векторів $W^{-1}B$. Позначаючи через r_0 кількість значень змінної класу C, матриця $W^{-1}B$ має щонайбільше $s = min(r_0 - 1, n)$ ненульових власних значень, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_s$, з відповідними лінійними незалежними власними векторами $v_1, v_2, \ldots, v_s$. Дискримінантні змінні визначаються як $u_l = xv_l$ і, зокрема,$w = v_l$ максимізує *w'Bw/w'Ww*.

Лінійний дискримінантний аналіз створює, для задачі двох класів, роздільну гіперплощину між двома наборами даних. Гіперплощина описується лінійною дискримінантною функцією $v_1x_1 + v_2x_2 + \cdots + v_nx_n + c$, яка дорівнює нулю на гіперплощині, якщо виконуються дві передумови: (a) багатовимірний нормальний розподіл в обох наборах даних і (b) однорідність обох коваріаційних матриць. Для дискримінантного аналізу гіперплощина визначається середнім геометричним між центроїдами (тобто центрами тяжіння) двох наборів даних. Щоб врахувати різні дисперсії та коваріації в наборах даних, змінні зазвичай спочатку перетворюються на стандартні середні ($\mu = 0$) і дисперсії ($\sigma^2 = 1$) і відстань Махаланобіса більш краща, ніж евклідова відстань.

Дерева класифікації - це інтуітивний спосіб класифікувати шаблон за допомогою послідовності запитань, у яких наступне запитання залежить від відповіді на поточне запитання. Також зазвичай відображають послідовність запитань у дереві спрямованої класифікації, яке також називають деревом класифікації, де кореневий вузол розташований у верхній частині, з'єднаний послідовними та спрямованими зв'язками або гілками з іншими вузлами. Вони так само з'єднуються, доки ми не досягнемо кінцевих або листових вузлів, які не мають подальших зв'язків. Класифікація конкретного шаблону починається з кореневого вузла, який запитує значення певної властивості шаблону. Різні посилання з кореневого вузла відповідають різним можливим значенням. Виходячи з відповіді, ми переходимо за відповідним посиланням на наступний або нащадковий вузол. У класифікаційних деревах зв'язки мають бути взаємно відмінними та вичерпними, тобто слідуватиметься одне й лише одне посилання. Наступним кроком є прийняття рішення у відповідному наступному вузлі, який можна вважати коренем піддерева. Ми продовжуємо цей шлях, доки не досягнемо кінцевого вузла, до якого більше не буде запитань. Кожен листовий вузол має мітку категорії, а тестовий шаблон призначається категорії досягнутого листового вузла.

Задачу створення моделі класифікаційного дерева з набору позначених даних можна розглядати як задачу організації змінних предикторів у дерево. Будь-яке дерево класифікації поступово розбиватиме набір навчальних даних

з мітками на все менші підмножини. В ідеальній ситуації всі зразки в кожній підмножині мали б однакову мітку категорії. У цій ситуації ми б сказали, що кожна підмножина була чистою і могла завершувати цю частину дерева.

Таким чином, для кожної гілки нам доведеться прийняти рішення або припинити розщеплення та прийняти недосконале рішення, або вибрати іншу змінну та розвивати дерево далі.

Найближчий сусід. Правило найближчого сусіда для класифікації x полягає в тому, щоб призначити його мітці, пов'язаній з прототипом, найближчим до контрольної точки:

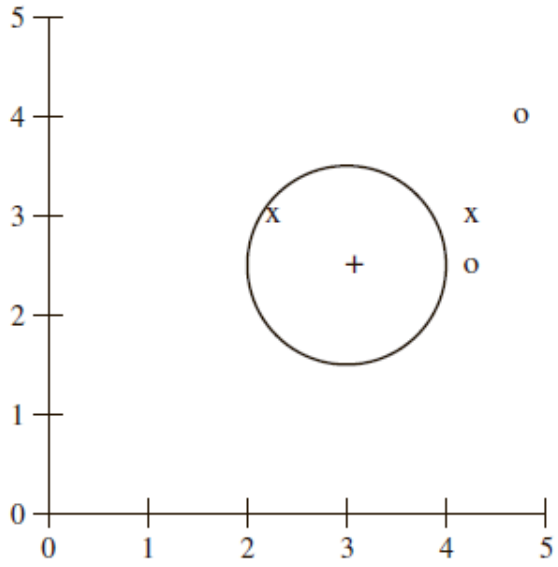


Рис.1.5. Класифікація найплижчого сусіда

Очевидним розширенням правила найближчого сусіда є правило k-найближчого сусіда. Це правило класифікує x, призначаючи його мітці, яка найчастіше представлена серед k найближчих зразків. Іншими словами, рішення приймається шляхом вивчення міток на k-найближчих-сусідів з подальшим вибором найоптимальнішої з них.

Практична проблема цього простого методу полягає в тому, що він має тенденцію працювати повільно для великих навчальних наборів, оскільки для кожного екземпляра тесту потрібно шукати весь набір. Стратегія уникнення обчислювальної складності алгоритму найближчого сусіда полягає в тому, щоб класифікувати кожен приклад відносно вже розглянутих прикладів і зберегти лише ті, які неправильно класифіковані. Ця стратегія відома як конденсація.

Слід зазначити, що ця парадигма не забезпечує явної моделі даних. Тому стверджується, що замість процесу індукції парадигма найближчого сусіда базується на процесі трансдукції, який уникає специфікації моделі.

Нейронні мережі. Штучні нейронні мережі виникли з ідеї математичного моделювання інтелектуальних здібностей людини за допомогою біологічно правдоподібних інженерних проектів. У штучній нейронній мережі елементарні блоки обробки (також звані «вузлами» або «нейронами») організовані в шари таким чином, що зазвичай з'єднані лише блоки, що належать до двох послідовних шарів. У структурі прямої нейронної мережі одиниця отримуватиме інформацію про декілька одиниць, що належать до попереднього рівня. Найпростіша нейронна мережа, яка називається персептрон, є однонейтронним класифікатором, який, використовуючи порогову функцію активації, розділяє два класи за допомогою лінійної дискримінаційної функції.

З'єднавши персептрони, ми можемо розробити структуру нейронної мережі під назвою багатошаровий персептрон. Це структура прямого зв'язку, оскільки вихідні дані вхідного рівня та всіх проміжних рівнів передаються лише на вищий рівень. Вектор ознак x подається на вхідний рівень, а на вихідний рівень – c, з дискримінантної функції g_1 (x),…,g_c (). Кількість прихованих шарів і кількість перцептронів на кожному прихованому шарі не обмежена. Можна показати, що багатошаровий персептрон з двома прихованими шарами порогових вузлів може апроксимувати будь-яку область класифікації з заданою точністю. У припущенні, що структура багатошарового персептрона вже обрана і зафіксована, задача визначення значень параметрів для всіх вузлів вирішується алгоритмом зворотного поширення.

Методии опорних векторів покладаються на попередню обробку даних для представлення шаблонів у високому вимірі — як правило, набагато вищому, ніж оригінальний простір ознак. З відповідним нелінійним відображенням у досить високий розмір, дані з двох категорій завжди можна розділити гіперплощиною. Цей вибір часто залежить від знання проблемної області. За відсутності такої інформації можна вибрати поліноми, гаусові чи інші базові функції. Розмірність картованого простору може бути як завгодно високою (хоча на практиці вона може бути обмежена обчислювальними ресурсами).

Визначаючи маржу як будь-яку позитивну відстань від гіперплощини рішення, метою навчання опорних векторних машин є знайти розділяючу гіперплощину з найбільшим запасом. Ми очікуємо, що чим більший запас, тим краще узагальнення класифікатора.

Опорні вектори (Рис 1.6) — це (перетворені) шаблони навчання, які (однаково) близькі до гіперплощини.

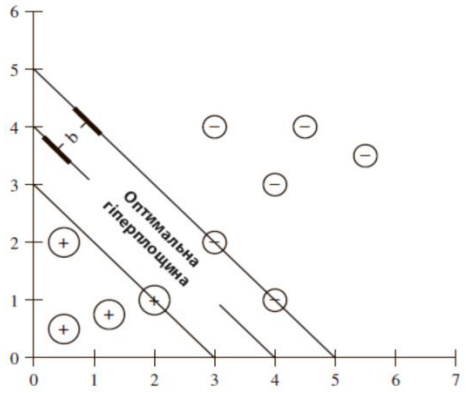


Рис. 1.6 Опорні вектори машинного класифікатора

Опорні вектори – це навчальні зразки, які визначають оптимальну роздільну гіперплощину, і є шаблонами, які найважче класифікувати. Неформально кажучи, вони є найбільш інформативними шаблонами для завдання класифікації.

Задача мінімізації величини вагового вектора, обмеженого поділом, може бути переформульована в необмежену задачу методом невизначених множників Лагранжа. Використовуючи так звану конструкцію Куна–Таккера, цю оптимізацію можна переписати як проблему максимізації, яку можна розв’язати за допомогою квадратичного програмування.

Об'єднання класифікаторів. Кожна парадигма класифікатора має пов'язану поверхню прийняття рішень. За допомогою комбінації класифікаторів можна отримати більш гнучкі поверхні прийняття рішень і більш точне рішення за рахунок збільшення складності. Комбінація класифікаторів — це сфера розпізнавання образів, яка швидко розвивається та привертає багато уваги спільноти машинного навчання. Важливим аспектом, який слід враховувати, є різноманітність різних базових класифікаторів, які потрібно комбінувати.

Поєднання класифікаторів може здійснюватися різними способами і на різних рівнях. Найпростіша стратегія — більшість голосів. Невидимий екземпляр буде класифіковано як клас, який отримує більше голосів від різних базових класифікаторів, вихідні мітки яких об’єднані. Подібні стратегії, які використовуються для об’єднання вихідних міток різних класифікаторів, — проста більшість (50%+1) і одноголосне голосування (усі згодні).

Іншим класичним способом комбінування різних базових класифікаторів є так зване стекове узагальнення. Ідея полягає в тому, щоб створити класифікатор з бази даних, що містить вихід класифікації кожного екземпляра початкової бази даних. Таким чином, кількість ознак, що характеризують екземпляри, збігається з кількістю базових класифікаторів.

Л. Брейман вводить концепцію bagging, як абревіатуру Bootstrap AGGregatING. Ідея пакетування проста та приваблива: ансамбль складається з класифікаторів, побудованих — з використанням унікального базового класифікатора — на початкових копіях навчального набору. Результати класифікатора об'єднуються більшістю голосів. Щоб використовувати варіації в навчальному наборі, базовий класифікатор повинен бути нестабільним, тобто невеликі зміни в навчальному наборі повинні призводити до великих змін у виході класифікатора. Одними з найбільш нестійких класифікаторів є дерева класифікації. Це пояснює пропозицію Бреймана під назвою випадкові ліси. Випадкові ліси — це загальний клас методів побудови ансамблю з використанням дерева класифікації як базового класифікатора. Іншим традиційним способом поєднання одного і того ж базового класифікатора є алгоритм AdaBoost. Термін походить від ADAptive BOOSTing. Загальна ідея полягає в тому, щоб поступово розвивати команду класифікаторів, додаючи по одному класифікатору. Класифікатор, який приєднується до ансамблю на одному кроці, навчається в наборі даних, вибірково відібраному з початкового набору даних навчання. Розподіл вибірки починається рівномірно та прогресує до збільшення ймовірності «складних» точок даних. Таким чином розподіл оновлюється на кожному кроці, збільшуючи ймовірність неправильної класифікації об'єктів на попередньому кроці.

У галузі статистики існує один підхід до моделювання, що називається байєсівським, який розглядає всі можливі структури та для кожної структури всі можливі значення параметрів. Це називається повним байєсівським підходом до моделювання. Це можна вважати крайнім випадком ансамблю класифікаторів лише з одним базовим класифікатором.

1.5. Керована класифікація в біоінформатиці

Геноміка. В проблемі пошуку генів використовувався вибір підмножини ознак. Наприклад, процедури оптимізації застосовуються до FSS у задачі прогнозування місця з'єднання. Опорні векторні машини та нейронні мережі використовуються для обчислювальної ідентифікації функціональних генів РНК. Класифікаційні дерева використовують для загальногеномної ідентифікації генів, які ймовірно залучені до генетичних захворювань, беручи різні показники збереження та довжину гена як прогнозні змінні.

Реконструкція амінокислотних послідовностей за допомогою спектральних ознак була розглянута за допомогою динамічного програмування. Динамічне програмування також є типом алгоритмів, яким надають перевагу для передбачення вторинної структури РНК. Для ідентифікації структурних елементів РНК використовувалися еволюційні

алгоритми. До визначення третинної структури РНК підійшли за допомогою пошуку табу.

Протеоміка. Кілька застосувань найближчого сусіда було зроблено для передбачення вторинної структури білків. Існує консенсусний метод, заснований на дереві класифікації для прогнозування вторинної структури білка.

С. Янг, Д. Добс та В Хонавар розробили двоетапний метод, що складається з машини опорних векторів і байєсівського класифікатора для прогнозування поверхневих залишків білка, які беруть участь у білок-білкових взаємодіях.

Проблема автоматичного прогнозування субклітинного розташування білка з його послідовності розглядалася за допомогою нечіткого алгоритму k-найближчого сусіда.

Мікрочіпи. Байєсівське узагальнення машини опорних векторів використовується для одночасного вибору оптимального класифікатора та оптимальної підмножини генів для діагностики раку на основі даних експресії. Також k-найближчий сусід використовується в поєднанні з генетичним алгоритмом у підході оболонки для відбору генів.

Системна біологія. Хоча імовірнісні графічні моделі є найбільш використовуваним підходом у системній біології, деколи всеж таки буває доцільно розглядати проблему з контрольованої точки зору. Наприклад, дерева класифікації використовують при моделюванні каскадів сигнал-відповідь, і методологій для прогнозування швидкості клітинної міграції за допомогою рівнів фосфорилювання сигнальних білків. Також використовують підсилення за допомогою дерев класифікації як базовий класифікатор для прогнозування регуляторної відповіді гена, яка вважається бінарною змінною (регульованою вгору або вниз).

1.6. Кластеризація

1.6.1. Вступ

Кластеризація полягає в розділенні набору елементів на підмножини відповідно до відмінностей між ними. Іншими словами, це процес групування подібних елементів. Основна відмінність від контрольованої класифікації полягає в тому, що в кластеризації ми не маємо інформації про кількість класів.

Найбільш типовим прикладом кластеризації в біоінформатиці є кластеризація генів у даних експресії. У нарисах мікрочипів ми отримуємо

значення експресії для тисяч генів у кількох зразках. Цінна інформація, яку ми можемо отримати з цих даних, полягає в тому, які гени спільно експресуються в різних зразках. Це проблема кластеризації, коли гени з однаковим рівнем експресії в усіх зразках групуються в кластер.

Кластерний аналіз, також званий сегментацією даних, має різноманітні цілі. Усі вони пов'язані з групуванням або сегментуванням колекції об'єктів на підмножини або «кластери», так що об'єкти в кожному кластері тісніше пов'язані один з одним, ніж об'єкти, призначені до різних кластерів. Іноді метою є впорядкування кластерів у природну ієрархію. Це передбачає послідовне групування самих кластерів таким чином, щоб на кожному рівні ієрархії кластери в одній групі були більш схожі один на одного, ніж у різних групах.

Центральним для всіх цілей кластерного аналізу є поняття ступеня подібності (або відмінності) між окремими об'єктами, що кластеризуються. Метод кластеризації намагається згрупувати об'єкти на основі наданого йому визначення подібності.

1.6.2. Підходи кластеризації

Кластеризація розділів (Рис. 1.7) спрямована на отримання розділу даних. Кожна точка належить до унікального кластера. Загальною стратегією є фіксація кількості кластерів, хоча деякі алгоритми можуть шукати найбільш відповідну кількість кластерів під час розподілу об'єктів у різних кластерах.

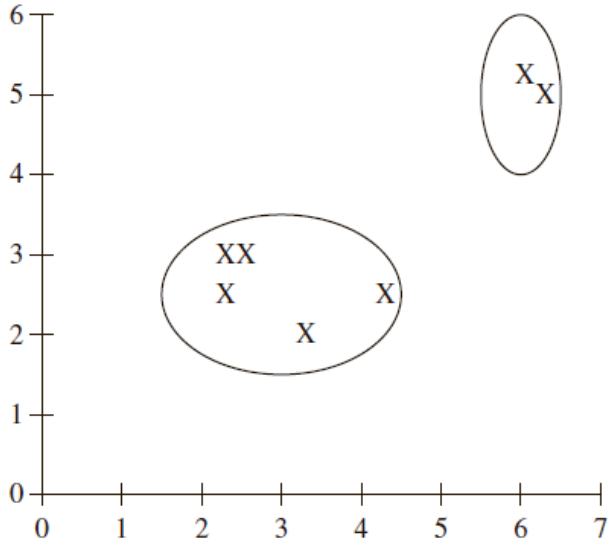


Рис. 1.7. Кластеризація розділів.

Алгоритм K-середніх є одним із найпопулярніших методів кластеризації з ітеративним спуском. Метою алгоритму K-середніх є розділення даних на K кластерів таким чином, щоб мінімізувати суму квадратів усередині групи. Найпростіша форма алгоритму K-середніх

заснована на чергуванні двох процедур. Перший — розподіл об’єктів по групах. Зазвичай об’єкт відносять до групи, середнє значення якої є найближчим у евклідовому розумінні. Друга процедура — розрахунок нових групових середніх на основі призначень. Процес завершується, коли жодне переміщення об’єкта до іншої групи не зменшить суму квадратів усередині групи.

Існує багато варіантів алгоритму K-середніх, які покращують його ефективність з точки зору скорочення часу обчислення та досягнення меншої помилки. Деякі алгоритми дозволяють створювати нові кластери та видаляти існуючі під час ітерацій. Інші можуть перемістити об’єкт до іншого кластера на основі найкращого вдосконалення цільової функції. В якості альтернативи можна використати перше покращення, яке було виявлено під час проходження набору даних.

Методом, пов’язаним з алгоритмом K-середніх, є векторне квантування. Основною метою векторного квантування є стиснення даних. Векторний квантувач складається з двох компонентів: кодера та декодера. Алгоритм, відомий як узагальнений алгоритм Ллойда, очевидно, є варіантом алгоритму K-середніх. Більше того, самоорганізуючі карти ознак є особливим типом векторного квантування, у якому існує впорядкування або топологія, накладена на кодові вектори. Метою самоорганізації є представлення високовимірних даних у вигляді низьковимірного масиву чисел (зазвичай у 1D або 2D масиві), який відображає структуру вихідних даних.

Ієрархічна кластеризація. Процедури ієрархічної кластеризації є найбільш часто використовуваними методами для узагальнення структур даних у біоінформатиці. Ієрархічне дерево (Рис 1.8) — це вкладений набір розділів, представлений деревоподібною діаграмою або дендрограмою. Розбиття дерева на певний рівень створює розбиття на K непересічних груп. Якщо дві групи вибрано з різних розділів (результат розділення на різних рівнях), то або групи є непересічними, або одна група повністю містить іншу. В ієрархічній кластеризації існує міра відстані або несхожості між двома об’єднаними кластерами. Матриця, що містить відмінності між парою кластерів, називається матрицею відмінностей. Ієрархічна структура будується шляхом злиття двох найближчих груп.

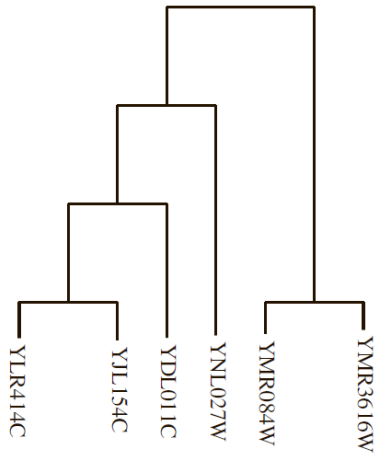


Рис 1.8. Приклад дендрограми, сформульованої ієрархічним методом

Існує кілька різних алгоритмів пошуку ієрархічного дерева. Агломеративний алгоритм починається з N підкластерів, кожен з яких містить одну точку, і на кожному етапі він об'єднує дві найбільш подібні групи, щоб утворити новий кластер, таким чином зменшуючи кількість кластерів на один. Алгоритм продовжується, поки всі дані не потраплять в один кластер. Алгоритм розділення працює шляхом послідовного поділу груп, починаючи з однієї групи і продовжуючи, доки не буде N груп, кожна з яких складається з однієї особи.

Найпоширенішими мірами відстані між кластерами є одинарний зв'язок (відстань між двома групами – це відстань між їхніми найближчими членами), повний зв'язок (визначається як відстань між двома найвіддаленішими точками), метод ієрархічної кластеризації Уорда (на кожному етапі алгоритму дві групи, які спричиняють найменше збільшення загальної суми квадратів усередині групи, об'єднуються), відстань до центроїда (визначається як відстань між середніми значеннями кластера або центроїдами), середня відстань (відстань між медіанами кластерів)) і середній груповий зв'язок (середнє значення відмінностей між усіма парами індивідуумів, по одній із кожної групи)[3].

*Моделі змішування.*У методі змішування кластеризації передбачається, що кожна окрема група в сукупності описується окремим розподілом ймовірностей. Популяція описується скінченним розподілом суміші у вигляді $p(x) = \sum_{i=1}^{K} \pi_i p(x;\, \theta_i)$, де π_i — пропорції змішування ($\sum_{i=1}^{K} \pi_i = 1$), а $p(x;\, \theta_i)$, — n-вимірна функція ймовірності, що залежить, у кожній суміші, від вектора параметрів θ_i. Є три набори параметрів для оцінки: значення π_i, компоненти векторів θ_i та значення K, кількість груп у сукупності.

Звичайний підхід до кластеризації з використанням кінцевих розподілів суміші полягає, перш за все, у визначенні форми розподілу компонентів, $p(x;\, \theta_i)$. Для неперервних змінних звичайним вибором є суміш

нормальних розподілів (кожен компонент слідує багатовимірному нормальному розподілу), тоді як для суміші бінарних змінних часто вибирають розподіл Бернуйлі. Після вказівки форми розподілу компонентів задається кількість кластерів *K*. Параметри моделі тепер оцінюються (це завдання може бути досягнуто за допомогою алгоритму ЕМ), а об'єкти групуються на основі їхніх оцінених апостеріорних ймовірностей приналежності до групи. Іншими словами, об'єкт *x* призначається до групи і, якщо $\pi_i p(x;\, \theta_i) \geq \pi_i p(x;\, \theta_j)$ для всіх $j \neq i, j = 1, \ldots, K$.

Основна складність у методі змішування стосується кількості компонентів *K*, яка майже в усіх підходах має бути визначена до того, як можна буде оцінити інші параметри. Інша проблема з підходом моделі змішування полягає в тому, що існує багато локальних мінімумів функції правдоподібності, і, можливо, доведеться спробувати кілька початкових конфігурацій, перш ніж буде створено задовільну кластеризацію.

Перевірка. Залежно від конкретного вибору методу попередньої обробки, вимірювання відстані, алгоритму кластеризації та інших параметрів, різні цикли кластеризації дадуть різні результати. Тому дуже важливо підтвердити релевантність кластера. Валідація може бути статистичною або біологічною. Статистичну перевірку кластерів можна здійснити шляхом оцінки узгодженості кластерів, шляхом вивчення передбачуваної потужності кластерів або тестування надійності кластерного результату щодо додавання шуму. З біологічної точки зору, дуже важко вибрати найкраще кластерне рішення, якщо біологічна система не була повністю охарактеризована.

Кластеризація в біоінформатиці. Основна галузь застосування методів кластеризації пов'язана з аналізом даних мікрочипів. Виходячи з припущення, що експресійна подібність (тобто спільна експресія) передбачає певну регуляторну чи функціональну подібність генів (і навпаки), проблема пошуку генів, які можуть бути залучені в той самий біологічний процес, перетворюється на задачу кластеризації генів у групи на основі їх подібності в профілях експресії.

Перше покоління алгоритмів кластеризації, застосованих до профілів експресії генів (K-середні, ієрархічна кластеризація та карти самоорганізації), були здебільшого розроблені поза біологічними дослідженнями. Хоча були отримані обнадійливі результати, деякі характеристики (такі як визначення кількості кластерів, кластеризація викидів і обчислювальна складність) часто ускладнюють їх використання для кластеризації даних виразу.

З цієї причини друге покоління алгоритмів кластеризації почало вирішувати деякі обмеження попередніх методів. Ці алгоритми включають, серед іншого, алгоритми на основі моделі самоорганізованого дерева;

алгоритми на основі якості, які створюють кластери з гарантією якості, яка гарантує, що всі члени кластера коекспресуються; і алгоритми бікластеризації.

1.7. Імовірнісні графічні моделі

Імовірнісні графічні моделі представляють багатовимірну спільну щільність ймовірності через добуток термінів, кожен з яких включає лише кілька змінних. Структура продукту представлена графіком, який пов'язує змінні, що з'являються в загальному терміні. Цей графік визначає форму розподілу продукту, а також надає інструменти для обґрунтування властивостей продукту. Хоча імовірнісні графічні моделі, які використовують неорієнтовані графи — мережі Маркова і регіональні апроксимації — також застосовуються в біоінформатиці, у цьому розділі ми обмежуємося імовірнісними графічними моделями, де відповідний граф є орієнтованим ациклічний граф. Ми розглядаємо два типи ймовірнісних графічних моделей залежно від статусу випадкових величин. Якщо всі змінні є дискретними, ми називаємо модель байєсівською мережею, тоді як у випадку безперервних змінних — відповідно до розподілу Гауса — ми представимо так звані мережі Гаусса.

Більш формально, нехай $X = (X_1, \dots, X_n)$ — вектор випадкових величин. Імовірнісна графічна модель для X — це графічна факторізація спільного узагальненого розподілу ймовірностей$p(X = x)$ (або просто $p(x)$). Представлення складається з двох компонентів: структури та набору локальних узагальнених розподілів ймовірностей. Структура S для X — це орієнтований ациклічний граф (DAG), який представляє набір умовних (не)залежнихтверджень щодо змінних X.

Структура S для X представляє твердження, що для всіх $\mathrm{i} = 1, \dots, \mathrm{n}, X_i$ та його не-нащадків існує незалежне pa_i^s, з урахуванням батьківських вузлів X_i для S. Таким чином, розкладання на множники виглядає наступним чином:$\rho(x) = \prod_{i=1}^{n} \rho(x_i | pa_i^s)$. Локальні узагальнені розподіли ймовірностей залежать від скінченного набору параметрів $\theta_s \in \Theta_s$. Таким чином, ми перепишемо попереднє рівняння наступним чином:$\rho(x|\theta_s) = \prod_{i=1}^{n} \rho(x_i | pa_i^s \theta_i)$, де $\theta_s = (\theta_1, \dots, \theta_n)$. Беручи до уваги обидві складові ймовірнісної графічної моделі, модель буде представленаяк $M = (S, \theta_s)$.

Імовірнісні графічні моделі можна використовувати для контрольованої класифікації, кластеризації та представлення зв'язків між різними змінними предметної області. У випадку кластеризації змінна, що позначає групу, вважається прихованою. Приховані моделі Маркова, дуже популярна парадигма в біоінформатиці, її можна розглядати як екземпляр імовірнісних графічних моделей (рис. 1.9). Щоб представити молекулярні

мережі за допомогою імовірнісних графічних моделей, ми пов'язуємо кожну молекулярну сутність із випадковою змінною. Значення цієї випадкової величини визначаються можливими рівнями молекулярної сутності. Ці типи стохастичних моделей виявилися дуже адекватними для представлення, наприклад, регуляції між генами.

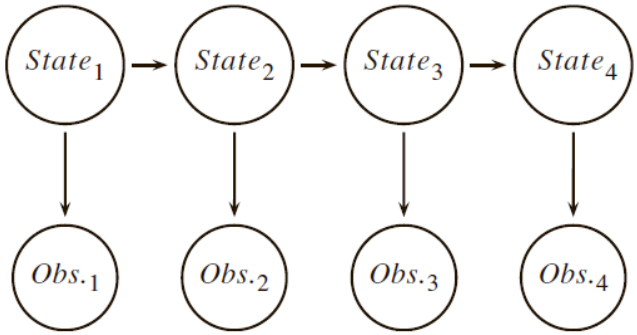


Рис 1.9. Структура прихованої моделі Маркова

Байєсівські мережі оточені підвищеною увагою в останні роки, про що свідчить велика кількість присвячених книг і широкий спектр теоретичних і практичних публікацій у цій галузі.

Парадигма байєсівської мережі в основному використовується для міркування в областях із внутрішньою невизначеністю. Байєсовські мережі використовуються для моделювання зв'язків між змінними. Бувають ситуації, коли значення деяких змінних системи відомі (це називається доказами), і нам може бути цікаво дізнатися, як ці докази впливають на розподіл ймовірностей решти змінних системи. Цей тип міркувань виконується за допомогою поширення доказів через байєсівську мережу, і це може бути доведено як NP-жорсткий ризик у загальному випадку багатозв'язних байєсівських мереж.

Коли байєсовська мережа побудована, вона стає ефективним пристроєм для виконання ймовірнісного висновку. Проте проблема побудови такої мережі залишається. Структура та умовні ймовірності, необхідні для характеристики байєсівської мережі, можуть бути надані експертами ззовні — це забирає багато часу та піддається помилкам — або шляхом автоматичного навчання з бази даних випадків. З іншого боку, завдання навчання можна розділити на дві підзадачі: вивчення структури, тобто визначення топології байєсівської мережі, і параметричне навчання, числові параметри (умовні ймовірності) для даної топології мережі.

Існує два основних способи вивчення байєсівських мереж з даних. Одним із них є виявлення умовної (не)залежності трійок змінних за допомогою перевірки гіпотез. Інший – це так званий метод оцінки – пошуку, який пояснюється далі.

Кожен алгоритм, який намагається відновити структуру байєсівської мережі шляхом виявлення (не)залежностей, має деякі умовні відносини (не)залежності між деякою підмножиною змінних моделі як вхідні дані та спрямований ациклічний граф, який представляє великий відсоток (і навіть усі вони, якщо це можливо) цих відносин як результат. Після вивчення структури умовні розподіли ймовірностей, необхідні для повної специфікації моделі, оцінюються з бази даних — з використанням деяких різних підходів до вивчення параметрів — або надаються експертом.

Хоча підхід до виявлення моделі, заснований на виявленні умовних (не)залежностей, є досить привабливим, через його близькість до семантики байєсівських мереж, великий відсоток розроблених алгоритмів навчання структури належить до категорії методів оцінки – пошуку. Щоб використовувати цей підхід до навчання, нам потрібно визначити метрику, яка вимірює якість кожної байєсівської мережі-кандидата щодо файлу даних випадків. Крім того, нам також потрібна процедура для розумного переміщення простором можливих мереж. У більшості підходів оцінки-пошуку пошук виконується в просторі орієнтованих ациклічних графів, які представляють можливі байєсовські мережеві структури. Інші можливості включають пошук у просторі класів еквівалентності байєсівських мереж або в просторі впорядкування змінних. Задача знаходження найкращої мережі за деяким критерієм із множини всіх мереж, у яких кожен вузол має не більше K батьків ($K > 1$), є NP-складною. Цей результат дає гарну можливість використовувати різні евристичні алгоритми пошуку. Ці евристичні методи пошуку можуть бути більш ефективними, коли критерій вибору моделі є роздільним, тобто, коли критерій вибору моделі можна записати як добуток (або суму) критеріїв, специфічних для змінної. Серед усіх евристичних стратегій пошуку, які використовуються для пошуку задовільних моделей у просторі байєсівських мережевих структур, у нас є різні альтернативи: жадібний пошук, імітований відпал, табуляційний пошук, генетичні алгоритми, еволюційне програмування, оцінка алгоритму розподілу тощо. Показники оцінки, які були які використовуються при навчанні байєсівських мереж на основі даних, штрафуються за максимальною правдоподібністю, за байєсівськими балами (наприклад, за граничною правдоподібністю) і за результатами теорії інформації.

Гаусові мережі. Інший окремий випадок імовірнісних графічних моделей, коли кожна одновимірна змінна X_i є неперервною, а кожна функція локальної щільності є моделлю лінійної регресії $f(x_i|pa_i^s\theta_i) \equiv \mathcal{N}\left(x_i; m_i + \Sigma_{x_j \in pa_i} b_{ji}(x_j - m_j), v_i\right)$ є однофакторним нормальним розподілом із середнім і дисперсією σ^2. Імовірнісна графічна модель, побудована за допомогою цих локальних функцій щільності, називається мережею Гауса.

Основна складність при роботі з багатовимірними нормальними розподілами полягає в тому, щоб переконатися, що оцінена коваріаційна матриця є позитивно-визначеною. Однак у представленні мережі Гаусса немає необхідності знати про це обмеження. Тому мережі Гауса більше підходять для виявлення та розуміння моделі, ніж стандартне представлення багатовимірних нормальних розподілів.

Як і у випадку з мережами Байєса, існують різні підходи до створення мереж Гауса на основі даних. Найбільш звичайні з них базуються на тестах виключення країв, штрафній метриці максимальної вірогідності та байєсівських балах.

Ймовірнісні графічні моделі в біоінформатиці

Геноміка. Основним застосуванням імовірнісних графічних моделей у геноміці є моделювання послідовностей ДНК. Приховані моделі Маркова використовуються в процесі пошуку генів, або в альтернативному виявленні сплайсингу. Генетичні алгоритми використовуються при навчанні прихованих марковських моделей для ідентифікації промоторних і кодуючих областей. Генне моделювання — не єдине застосування імовірнісних графічних моделей. Наприклад, байєсовські мережі використовуються при моделюванні блоків гаплотипів, а пізніше ці моделі використовуються для картування нерівноважності зв'язку.

Протеоміка. Мережі Байєса використовувалися для прогнозування карт контакту білків і для проблеми розпізнавання складок білків і класифікації надродини.

Системна біологія. Одним із найважливіших застосувань імовірнісних графічних моделей є висновок генетичних мереж.

Деякі переваги використання цієї парадигми для моделювання генетичних мереж ґрунтуються на теорії ймовірностей, науковій дисципліні, що має серйозний математичний розвиток. Теорія ймовірностей може бути використана як основа для боротьби з невизначеністю та шумом, що лежить в основі біологічних областей. Графічний компонент цих моделей — структура — дозволяє представити взаємозв'язки між генами — змінними — у спосіб, який можна інтерпретувати. Умовна незалежність між трійками змінних дає чітку семантику. Кількісна частина моделей — умовні ймовірності — дозволяє встановити силу взаємозалежностей між змінними. Алгоритми логічного висновку — точні та наближені — розроблені в цих моделях, дозволяють використовувати різні типи міркувань усередині моделі. Вже існують алгоритми, які шукають імовірнісні графічні моделі на основі даних спостережень на основі добре зрозумілих принципів статистики. Ці алгоритми дозволяють включати приховані змінні, які не

спостерігаються в реальності. Також можна об’єднати декілька локальних моделей у спільну глобальну модель. Декларативний характер імовірнісних графічних моделей є перевагою процесу моделювання завдяки врахуванню додаткових аспектів, таких як існування деяких ребер у моделі на основі попередніх знань. Моделі піддаються біологічній інтерпретації та можуть бути строго оцінені за даними спостережень[2].

Однак не всі характеристики імовірнісних графічних моделей підходять для цього завдання. Недоліком є те, що було зроблено дуже мало роботи щодо розробки алгоритмів навчання, здатних відобразити причинно-наслідковий зв’язок між змінними. Опис випадкових зв’язків між рівнями експресії генів є особливо важливим для отримання біологічного розуміння механізмів, що лежать в основі клітини. Крім того, особливості проаналізованих баз даних з дуже малою кількістю випадків, порядку десятків, і дуже великою кількістю змінних, порядку тисяч, роблять необхідним адаптувати розроблені алгоритми навчання. Таким чином, алгоритми навчання, які здатні виконувати моделювання підмереж і, в той же час, забезпечують надійність отриманої графічної структури, повинні представляти інтерес. Нарешті, включення прихованих змінних — де і скільки — є важкою проблемою під час вивчення імовірнісних графічних моделей на основі даних.

У науковій літературі були запропоновані статичні та динамічні ймовірнісні графічні моделі для реконструкції мереж експресії генів на основі даних мікрочипів.

Інформація про послідовність ДНК змішується з даними мікрочипу в байєсівській мережі, щоб отримати більш точну оцінку мережі, коли кількість даних мікрочипу обмежена. Генетичні мережі, оцінені на основі даних експресії, уточнюються за допомогою білок-білкових взаємодій. С. Імото пропонує новий метод вимірювання надійності передбачуваних генетичних мереж на основі початкового завантаження. Статичні мережі Гауса також були запропоновані для висновку про генетичні регуляторні мережі.

Динамічні байєсівські мережі здатні показати, як гени регулюють один одного в часі в складній роботі регуляторних шляхів. Аналіз даних часових рядів потенційно дозволяє нам визначити регуляторні шляхи в часі, а не просто асоціювати гени, які регулюються разом. У різних роботах розглядалося використання динамічних байєсівських мереж для визначення регуляторних шляхів.

Методи кластеризації, застосовані до даних мікрочіпів і даних взаємодії білок-білок, поєднуються в побудові мережі передачі сигналу.

1.8. Оптимізація

Багато проблем у біоінформатиці можна поставити як задачу пошуку оптимального рішення в просторі кількох (іноді експоненціальних) можливих рішень. Вибір методу оптимізації, який буде використано, є вирішальним для вирішення проблеми. У цьому розділі ми описуємо кілька алгоритмів оптимізації, розроблених спільнотою машинного навчання, і розглядаємо їх застосування до проблем біоінформатики.

У нашому аналізі ми не будемо розглядати низку класичних методів оптимізації та евристичних методів, які, хоча й широко використовуються для вирішення біологічних проблем, не є актуальними з точки зору машинного навчання. Ці методи включають підйом на пагорб, жадібну евристику, динамічне та цілочисельне програмування, а також методи розгалужень і зв’язків. Однак у розділі, в якому розглядаються оптимізаційні програми для біоінформатики, ми включаємо посилання на використання цих класичних методів оптимізації, щоб проілюструвати різні альтернативи розглянутим проблемам.

Оптимізаційні підходи до задач біоінформатики можна класифікувати, залежно від типу знайдених рішень, на точні та наближені методи. Точні методи виводять точні рішення, коли досягається збіжність. Однак вони не обов'язково збігаються в кожному випадку. Наближені алгоритми завжди виводять варіант рішення, але не обов’язково оптимальне.

Методи точної оптимізації. Загальні підходи до точної оптимізації включають методи вичерпного пошуку. Однак ці алгоритми можливі лише для невеликих пошукових доменів і не стосуються нашого огляду. Деякі методи здатні використовувати знання про проблему, щоб зменшити простір пошуку. Це можна зробити шляхом дотримання деяких обмежень, яким має відповідати оптимальне рішення.

Методи наближеної оптимізації. Наближені алгоритми можна далі класифікувати на детерміновані та стохастичні, відповідно до способу знаходження рішень. За наявності набору вхідних параметрів детермінований метод буде сходитися до того самого рішення. У стохастичних методах використовується випадковий компонент, який може призвести до отримання різних рішень при роботі з однаковими вхідними параметрами.

Стохастичні алгоритми можна розділити на локальні та популяційні методи пошуку. Алгоритми локального пошуку відвідують одну точку простору пошуку на кожній ітерації. Методи пошуку на основі сукупності використовують набір або сукупність точок замість однієї точки. Прикладами локальних методів пошуку є пошук Монте-Карло, імітований відпал і табуляційний пошук.

При використанні в системі оптимізації алгоритм Монте-Карло пов’язує розподіл ймовірностей з кожною точкою простору пошуку на основі цільової функції. Ланцюг Монте-Карло створює ланцюг Маркова конформацій, який за досить велику кількість ітерацій наближає канонічний розподіл. Конфігурації, отримані методом, є зразками з простору пошуку і можуть бути об’єднані з мінімізацією енергії для пошуку оптимального рішення.

Імітація відпалу заснована на процесі відпалу, який виникає у фізиці. Він використовує ймовірності переходу на основі розподілу Больцмана та незростаючу функцію, яка називається графіком охолодження, для налаштування пошуку оптимальних рішень. Пошук за табуляцією дозволяє евристичним алгоритмам локального пошуку уникнути локальних мінімумів, коли алгоритм не може знайти жодного подальшого рішення в околицях, яке покращує значення цільової функції. Загальний підхід полягає в тому, щоб уникнути введення циклів, забороняючи або штрафуючи рухи, які алгоритм робить у наступній ітерації до точок у просторі рішень, які відвідали раніше.

Еволюційні алгоритми є одними з найвідоміших методів пошуку на основі популяції. Вони починаються із випадкової популяції точок і повторюють, доки не буде виконано певний попередньо визначений критерій зупинки. На кожній ітерації, яка зазвичай називається генерацією, вибирається підмножина точок. Застосовуючи деякі оператори варіації до вибраного набору, створюється нова сукупність. Прикладом еволюційних алгоритмів є генетичні алгоритми (ГА). Відмінною рисою ГА є застосування операторів рекомбінації та мутації. Іншим еволюційним алгоритмом, який використовується для вирішення біоінформаційних проблем, є генетичне програмування, яке використовується для розробки програмного коду, здатного вирішити дану проблему. Інший клас методів пошуку на основі сукупності включає ті алгоритми, які використовують імовірнісне моделювання рішень замість генетичних операторів. Алгоритми оцінювання розподілу (EDA) — це еволюційні алгоритми, які будують явну ймовірнісну модель набору вибраних рішень. Ця модель може фіксувати за допомогою імовірнісних залежностей релевантні взаємодії між змінними проблеми. Модель зручно використовувати для створення нових перспективних рішень.

Оптимізація в біоінформатиці

Геноміка. Було запропоновано кілька алгоритмів оптимізації для вирішення проблеми вирівнювання множинної послідовності (рис. 1.10). До них відносяться табуляційний пошук, оптимізація Монте-Карло, методи, засновані на генетичних алгоритмах, методи релаксації, імітований відпал, ітераційні алгоритми і паралельний імітований відпал.

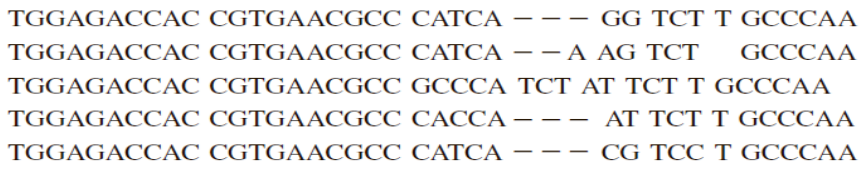


Рис. 1.10. Множинне вирівнювання послідовностей ДНК

Передбачення промоторів з послідовностей ДНК було досягнуто за допомогою ГА разом із нейронними мережами. Для відновлення оперонної структури геному прокаріотів було застосовано нечітко керований Га. Розвинуті нейронні мережі також показали хороші результати для завдання розрізнення функціональних елементів пов'язаних з кодуючими нуклеотидами з некодуючих послідовностей ДНК. Оптимізація архітектури нейронної мережі за допомогою генетичного програмування покращила виявлення та моделювання ген-генних взаємодій у дослідженнях захворювань людини. Крім того, оцінка алгоритмів розподілу була застосована для прогнозування місця сплайсингу і відбору генів.

До секвенування ДНК підходили за допомогою табуляційного пошуку, ГА і жадібних алгоритмів. Табуляційний пошук нещодавно також використовувався для визначення послідовностей амінокислот у довгих поліпептидах і для виділення мотивів із послідовностей ДНК.

Фізичне відображення хромосом було оброблено за допомогою методів оптимізації розгалужень і меж, алгоритмів Монте-Карло, жадібних методів і паралельних ГА. Ідентифікація консенсусної послідовності на послідовностях ДНК була використана за допомогою методів лінійного програмування та імітованого відпалу. Розбиття блоків гаплотипів і вибір SNP тегів оброблялися за допомогою алгоритмів динамічного програмування.

Реконструкція амінокислотних послідовностей з використанням лише спектральних ознак була вирішена за допомогою динамічного програмування. Динамічне програмування також є кращим вибором для передбачення вторинної структури РНК, яке, загалом, можна обробляти за допомогою поліноміальних алгоритмів. Еволюційні алгоритми також використовувалися для виявлення структурних елементів РНК. До визначення третинної структури РНК підійшли за допомогою табуляційного пошуку.

Протеоміка. Для згортання білка в спрощених моделях було використано кілька підходів до оптимізації. До них належать: табуляційний пошук, методи Монте-Карло та ГА.

Передбачення бічного ланцюга білка, важлива задача для прогнозування структури білка на основі гомології та дизайну білка,

розглядалося з використанням тупикових алгоритмів усунення, ГА та інших методів пошуку на основі популяції. Імітований відпал, методи оптимізації на основі висновків із графічних моделей і підхід самоузгодженого середнього поля також використовувалися для вирішення цієї проблеми.

Імітований відпал використовувався для моделювання петель у білкових структурах, а генетичне програмування використовувалося для прогнозування контактної карти в білках.

Системна біологія. Існує кілька застосувань генетичного програмування для висновку про генні мережі і метаболічні шляхи на основі спостережених даних. Ідентифікація сайтів зв'язування факторів транскрипції була оброблена за допомогою оптимізації ланцюга Маркова.

ГА були застосовані для моделювання генетичних мереж, вибору регуляторних структур та оцінки параметрів біопроцесів. Виведення генетичних мереж було досягнуто за допомогою інших еволюційних алгоритмів.

Мікрочіпи. Моделювання відпалу нещодавно було застосовано для розробки двоканальних досліджень мікрочипів. Його також використовували для узгодження експериментальних профілів транскрипції з набором еталонних експериментів, бікластеризації даних експресії та в аналізі тимчасових профілів експресії генів.

Еволюційні алгоритми були використані для кластеризації даних мікрочипів. ГА також застосовувалися для нормалізації даних експресії генів, що є необхідним кроком перед квантуванням даних експресії генів у бінарний домен. Багатокласове передбачення даних експресії генів було виконано за допомогою ГА. Для аналізу даних експресії генів були застосовані методи *k*-найближчих генетичних гібридів.

Еволюція та інші програми. Висновок про найкраще філогенетичне дерево, яке відповідає даним, було зроблено за допомогою різних методів оптимізації. Вичерпаний пошук використовувся, коли розмір простору пошуку малий. Розгалуження та зв’язування та інші евристичні методи були застосовані в інших випадках.

Використовувалися жадібні алгоритми, методи сходження на пагорб і симуляція відпалу, враховуючи їх просту та швидку реалізацію. До реконструкції гаплотипу підходили як з використанням точних, так і наближених методів. Проблеми малого розміру вирішувалися за допомогою методів розгалуження та зв’язування, тоді як більш складні випадки розв’язувалися за допомогою ГА.

Генетичні алгоритми використовувалися для оптимізації досліджень нерівноважності зчеплення, мінімізації тягаря генотипування, зворотного переходу послідовності білка в послідовність нуклеїдної кислоти і для розробки праймерів. Для покращення фрактальної візуалізації даних послідовності також використовувалися еволюційні алгоритми.

Роздід 2. Лінійна модель та її розширення

2.1.Метод найменших квадратів для простої лінійної регресії

Проста лінійна регресія – це комбінація моделей середнього та нахилу

$$Y_i = \alpha + \beta x_i + \varepsilon_i, \quad i = 1, 2, ..., n,$$

де x_i – фіксоване значення, яке іноді називають незалежною змінною або предиктором (або коваріативною величиною), α – відрізок, β — нахил, що підлягає оцінці, ε_i - неспостережувана помилка з нульовим середнім і постійною дисперсією σ^2. Зауважте, що Y у верхньому регістрі означає, що залежна змінна є випадковою, а x у нижньому регістрі означає, що незалежна змінна є фіксованою та відомою. Крім того, припускається, що $\{x_i\}$ приймє принаймні два різних значення або еквівалентно

$$\sum_{i=1}^{n}(x_i - \overline{x})^2 \neq 0. \tag{2.1}$$

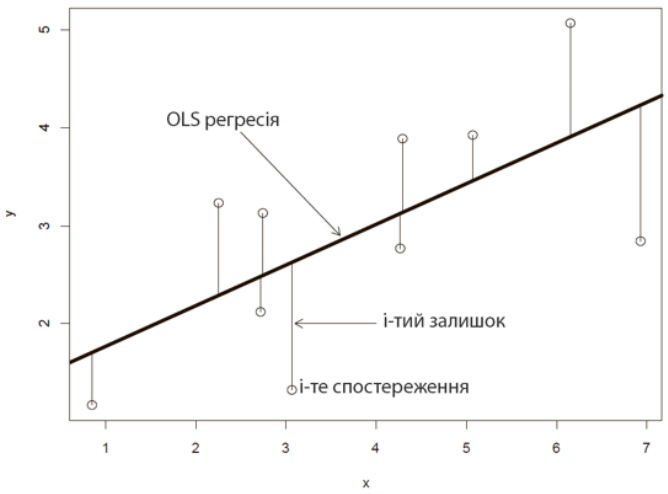
Тут припускається, що умови похибки ε_i і ε_j, не корелюють ($i \neq j$). Оскільки $\alpha + \beta x_i$ є невипадковими, Y_i та Y_j також не корелюють.В іншому формулюванні модель простої лінійної регресії може бути записана як

$$E(Y_i) = \alpha + \beta x_i.$$

Припущення, що ε_i має нульове середнє, не є дуже важливим: справді, якщо $E(\varepsilon_i) = \nu$, тоді ми можемо поєднати α і ν і переписати модель як $Y_i = \alpha' + \beta x_i + \varepsilon'_i$ де $\alpha' = \alpha + \nu$, а $\varepsilon'_i = \varepsilon_i - \nu$ з $E(\varepsilon'_i) = 0$. Однак ν і початкове відсікання α не можна ідентифікувати одночасно, якщо ν невідоме, можна ідентифікувати лише суму $\alpha + \nu$.

Звичайні оцінки найменших квадратів (OLS) перетину та нахилу в простій лінійній регресії мінімізують залишкову суму квадратів (RSS)

$$S(\alpha, \beta) \overset{\text{def}}{=} \sum_{i=1}^{n}(Y_i - \alpha - \beta x_i)^2.$$



Геометрична ілюстрація OLS. Відстань між спостереженнями і лінія регресії вздовж осі y.

Рис. 2.1. Геометрична ілюстрація OLS

Слід зауважити, що що сума квадратів не зважена через гомоскедастичні помилки. Відстань між і-тим спостереженням і лінією регресії вздовж *y*-осі називається і-тим залишком,Y_i - α -βx_iтому ми називаємо *S* залишковою сумою квадратів. Щоб знайти рішення за методом найменших квадратів, ми диференціюємо *S* відносно α таβ і прирівнюємо похідні до нуля,

$$\frac{\partial S}{\partial \alpha} = -2\sum_{i=1}^{n}(Y_i - \alpha - \beta x_i) = 0, \quad \frac{\partial S}{\partial \beta} = -2\sum_{i=1}^{n}(Y_i - \alpha - \beta x_i)x_i = 0.$$

Ці рівняння є необхідними умовами (пізніше буде доведено, що вони також є достатніми умовами) мінімуму залишкової суми квадратів і називаються нормальними рівняннями. Перепишемо ці рівняння у вигляді системи двох лінійних рівнянь:

$$\alpha n + \beta\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} Y_i, \quad \alpha\sum_{i=1}^{n} x_i + \beta\sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i Y_i.$$

Можна застосувати правило Крамера, щоб розв’язати цю систему для невідомого відрізка та нахилу, що дає розв’язок для оцінок OLS:

$$\widehat{\beta} = \frac{\sum_{i=1}^{n} x_i Y_i - n\overline{x}\,\overline{Y}}{\sum_{i=1}^{n} x_i^2 - n\overline{x}^2}, \quad \widehat{\alpha} = \overline{Y} - \widehat{\beta}\overline{x}. \tag{2.2}$$

Аналогічно, коефіцієнт нахилу можна обчислити як

$$\widehat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(Y_i - \overline{Y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}. \tag{2.3}$$

Завдяки припущенню (1) розв’язок нормального рівняння існує і єдиний.

Вище наведений малюнок ілюструє OLS: кола представляють точки даних (x_i, Y_i), а жирна лінія зображує лінію регресії. Відстань від точок даних до лінії регресії вздовж осі y зображено сегментами. і-й залишок $r_i = Y_i - \hat{\alpha} - \hat{\beta}x_i$. Для лінії регресії OLS ми маємо $\min\sum_{i=1}^{n} r_i^2$. Той факт, що відстані вимірюються вздовж осі y, відповідає тиму фактору, що залежна змінна є випадковою, але предиктор фіксований. Наприклад, якби X було випадковим, але Y фіксованим, відстань вимірювалася б уздовж осі x. Якби X і Y були випадковими з однаковими дисперсіями, відстань між точкою (X, Y) і лінією була б мінімальним квадратом довжини перпендикуляра, опущеного з (X, Y) на лінію. Спосіб обчислення відстані відображає припущення про випадковість змінних.

Лінія регресії методу найменших квадратів проходить через центр даних $(\bar{x}, \bar{Y})$. Тоді , $\bar{Y} = \hat{\alpha} + \hat{\beta}\bar{x}$. Отже, з другого рівняння (2) ми отримуємо $\hat{\alpha} + \hat{\beta}\bar{x} = \bar{Y}$ - $\hat{\beta}\bar{x} + \hat{\beta}\bar{x} = \bar{Y}$

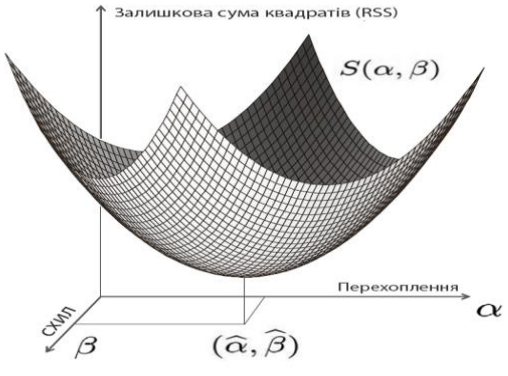
Функція S є квадратичною, строго опуклою функцією від αіβ, оскільки матриця Гессе,

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 S}{\partial \alpha^2} & \frac{\partial^2 S}{\partial \alpha \partial \beta} \\ \frac{\partial^2 S}{\partial \beta \partial \alpha} & \frac{\partial^2 S}{\partial \beta^2} \end{bmatrix} = 2 \begin{bmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{bmatrix},$$

є додатно визначеною. Дійсно, ця матриця є додатно визначеною, оскільки елементи на головній діагоналі додатні, а визначник додатний,

$$n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2 = n\sum_{i=1}^{n}(x_i - \overline{x})^2 > 0,$$

Оскільки функція S є строго опуклою функцією, нормальні рівняння є достатніми умовами для мінімуму, і мінімум досягається в єдиній точці з координатами, заданими в рівнянні (2). Типова функція RSS зображена на малюнку нижче: коли α абоβ прагне до нескінченності, S також прагне до нескінченності (площина піднімається).



Мінімізація RSS в якості функції α та β. Оскільки S(α, β) є строго опуклою функцією, то рішення методу найменших квадратів існує і є унікальним.

Рис. 2.2. Мінімізація

2.2.Статистичні властивості оцінки OLS за нормальним припущенням

Вишченаведені згадуванняOLS не вимагали припущення щодо розподілу помилок ε_i. Як ми побачимо пізніше, OLS стає найкращим методом оцінки, якщо помилки розподілені нормально. Наступний результат є орієнтиром лінійних моделей. Він формулює оптимальні властивості МНК і основних розподілів для простої лінійної регресії.

Теорема про статистичні властивості оцінки OLS. *Якщо* ε_i *має нормальний розподіл, тоді*

(a) OLS оцінки перетину,$\hat{\alpha}$*, і нахилу,* $\hat{\beta}$*, мають мінімальні дисперсії серед усіх незміщених оцінок (покращена теорема Гаусса-Маркова).*

(b) $\hat{\alpha}$*та*$\hat{\beta}$ *мають двовимірний нормальний розподіл із граничними розподілами, заданими як*

$$\widehat{\alpha} \sim \mathcal{N}\left(\alpha, \sigma^2 \frac{\sum_{i=1}^n x_i^2}{n\sum_{i=1}^n (x_i-\overline{x})^2}\right), \quad \widehat{\beta} \sim \mathcal{N}\left(\beta, \sigma^2 \frac{1}{\sum_{i=1}^n (x_i-\overline{x})^2}\right).$$

(c) Оцінювач

$$\widehat{\sigma}^2 = \frac{1}{n-2}\sum_{i=1}^n (Y_i - \widehat{\alpha} - \widehat{\beta} x_i)^2$$

є незміщеним для σ^2*, а нормалізована сума квадратів має квадратичний розподіл:*

$$(n-2)\frac{\widehat{\sigma}^2}{\sigma^2} \sim \chi^2(n-2).$$

(d) $\hat{\beta}$ *мінус його справжнє значення, поділене на стандартну помилку, має t-розподіл:*

$$\frac{\widehat{\beta}-\beta}{\widehat{\sigma}/\sqrt{\sum_{i=1}^n (x_i-\overline{x})^2}} \sim t(n-2).$$

Функція lm і прогнозування за допомогою лінійної регресії

Оцінка лінійної регресії за допомогою OLS є частим завданням у багатьох статистичних програмах. У R обчислення та відповідний вихід статистичної інформації забезпечується вбудованою функцією lm. Ця функція також працює для простої лінійної регресії та множинної регресії з перехопленням або без нього. Якщо y і x є масивами однакової довжини, ми видаємо lm(y~x) для виконання простої лінійної регресії. Перехоплення наявне за замовчуванням. Щоб оцінити модель нахилу без відрізка ми пишемо lm(y~x-1). Іноді залежні та незалежні змінні є частиною кадру даних.

Потім ми посилаємося на змінні за їхніми іменами та вказуємо фрейм даних за допомогою опції даних. Наприклад, якщо ім'я кадру даних — datfr, а імена залежної та незалежної змінних — ydep і xind відповідно, викликаємо — lm(ydep~xdp,data=datfr).

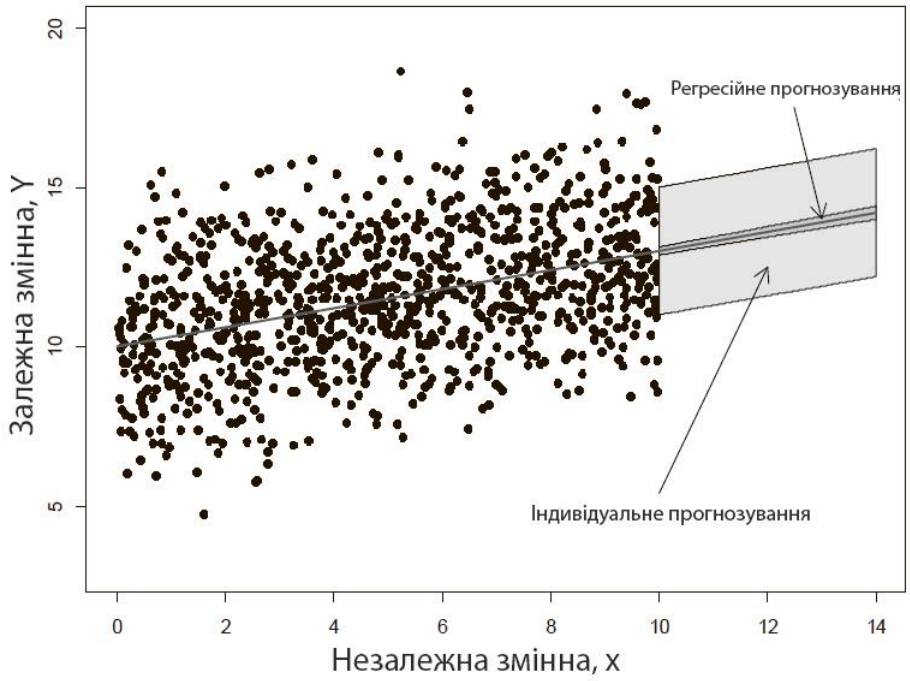


Рис.2.3. Різниця між індивідуальним і регресійним прогнозуванням

Для індивідуального прогнозу $Y = \alpha + \beta x + \varepsilon$, але регресійний прогноз передбачає точку на лінії регресії,$\alpha + \beta x$. Значення прогнозу те саме, але перше має дисперсію більшу на σ^2.

Однією з найпривабливіших особливостей лінійної регресії є можливість передбачити залежну змінну за значенням незалежної змінної[4]. Ми розрізняємо два типи прогнозу: регресійний/середній прогноз та індивідуальний прогноз. Якщо задано значення x, спочатку прогнозується регресійне значення $\alpha + \beta x$, а потім прогнозується$\alpha + \beta x + \varepsilon$. В обох випадках прогноз є однаковим, $\hat{\alpha} + \hat{\beta} x$, але дисперсія на σ^2 більша для індивідуального прогнозу. Щоб підкреслити різницю, ми позначаємо $\hat{Y} = \hat{\alpha} + \hat{\beta} x$ як прогноз регресії та $\tilde{Y} = \hat{\alpha} + \hat{\beta} x + \varepsilon$ як індивідуальний прогноз, де ε не залежить від $\{Y_i, i = 1, \dots, n\}$

Дано

$$\frac{\widehat{Y} - \alpha - \beta x}{\widehat{\sigma}\sqrt{q(x)}} \sim t(n-2), \quad \frac{\widetilde{Y} - \alpha - \beta x}{\widehat{\sigma}\sqrt{1+q(x)}} \sim t(n-2), \tag{2.4}$$

де $q(x) = 1/n + (\overline{x} - x)^2 / \sum (x_i - \overline{x})^2$.

Тут ми можемо побудувати точний інтервал для індивідуального прогнозу:

$$\Pr\left(\widehat{Y} - t_{1-\alpha/2}(n-2)\widehat{\sigma}\sqrt{1+q(x)} < Y < \widehat{Y} + t_{1-\alpha/2}(n-2)\widehat{\sigma}\sqrt{1+q(x)}\right) = 1-\alpha,$$

де це (1 -)-ий квантиль t-розподілу з n − 2 ступенями свободи Ми інтерпретуємо цей вираз, кажучи, що двосторонній інтервал покриває майбутнє Y з імовірністю 1 –α. Якщо нас цікавить односторонній інтервал, скажімо,

$$\Pr\left(Y > \widehat{Y} + t_{\alpha}(n-2)\widehat{\sigma}\sqrt{1+q(x)}\right) = 1-\alpha.$$

Можна використовувати α = 0.05, тоді ми отримуємо 95% інтервал довіри.

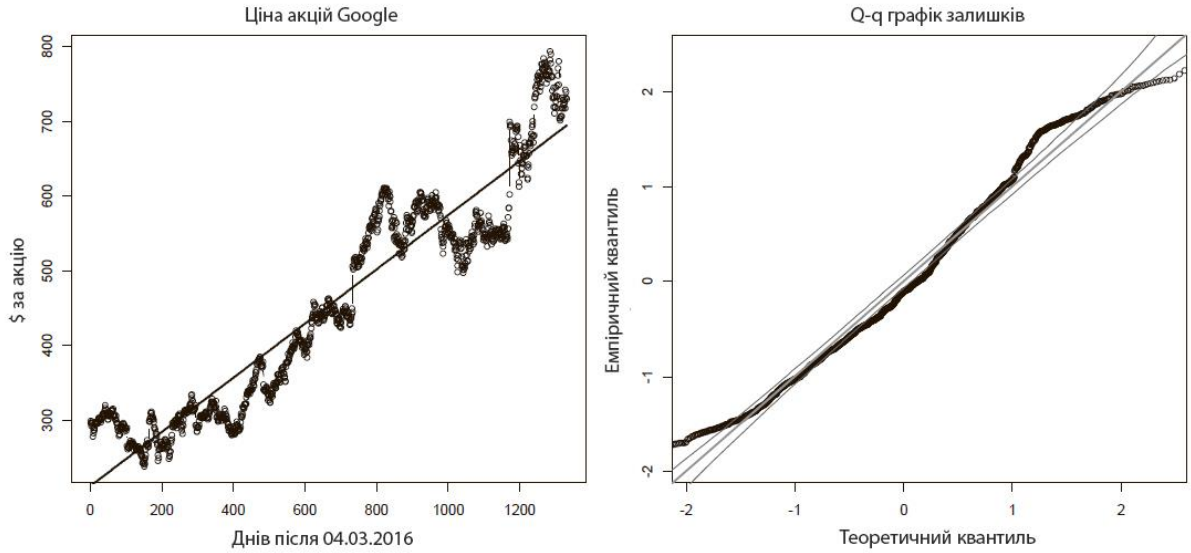


Рис2.4. Ціна акцій Google із підігнаним трендом OLS і графіком q-q залишків.

Приклад. Прогноз курсу акцій Google. (a) Оцінити тенденцію щоденної ціни акцій Google, починаючи з 19 серпня 2004 року до 4 березня 2016 року, (b) побудувати графік q-q, щоб перевірити, чи залишки мають нормальний розподіл, і (c) передбачити акції ціни з 05.12.2016 по 03.01.2017.

Розв'язання. (a) Код R знаходиться у функції lm.trendSP. Підсумок результату lm наведено нижче.

```
> lm.trendSP(job=1)
Call:
lm(formula = y ~ x)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 2.118e+02  2.619e+00   80.85   <2e-16 ***
x           3.628e-01  3.407e-03  106.49   <2e-16 ***
Residual standard error: 47.75 on 1329 degrees of freedom
Multiple R-squared:  0.8951,    Adjusted R-squared:  0.895
F-statistic: 1.134e+04 on 1 and 1329 DF,  p-value: < 2.2e-16
```

Перший рядок показує виклик функції. Другий рядок містить підсумкову статистику для залишків, $Y_i - \hat{\alpha} - \hat{\beta}x_i$: мінімальні значення,

перший квартиль, медіана, третій квартиль і максимум. Таблиця коефіцієнтів є найважливішою частиною результату. У ньому наведено оцінки OLS перетину та нахилу, їхні стандартні помилки, обчислені за формулами з теореми Гауса—Маркова із σ, заміненим його оцінкою $\sqrt{\hat{\sigma}^2}$, t-статистикою як відношенням оцінок до стандартної помилки, а також p-значення Pr (>|t|). Залишкова стандартна помилка дорівнює $\hat{\sigma}$, ступені свободи дорівнюють *n* мінус кількість оцінених коефіцієнтів ($n - 2$). Кратний R-квадрат обчислюється за допомогою

$$R^2 = 1 - \frac{\sum_{i=1}^{n} r_i^2}{\sum_{i=1}^{n} (Y_i - \overline{Y})^2}, \quad (5)$$

а скоригований R-квадрат обчислюється за формулою

$$R^2_{\text{adj}} = 1 - \frac{\sum_{i=1}^{n} r_i^2/(n-2)}{\sum_{i=1}^{n} (Y_i - \overline{Y})^2/(n-1)}.$$

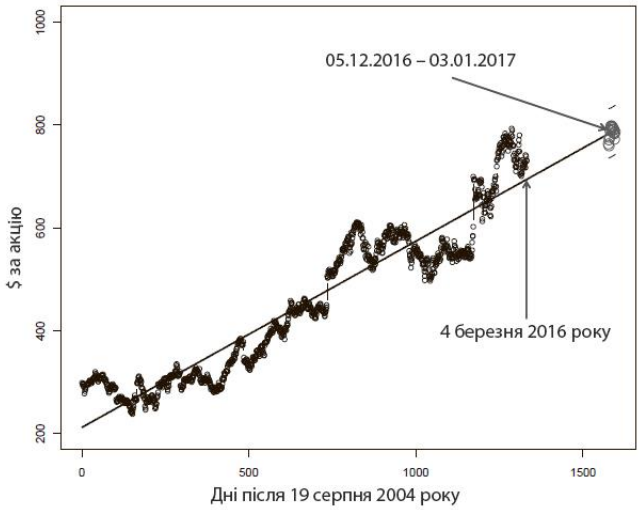


Рис. 2.5. Прогнозування цін на акції Google за лінійним трендом.

(b) Цінові дані та підібрана лінія регресії разом із графіком q-q показані на на малюнку (1) Ми використовуємо параметр type="b" на графіку, щоб показати точки, з'єднані відрізками. Діапазон довіри 95% показано на графіку q-q. Лінія посередині — це лінія 45◦. Точки знаходяться за межами довірчого діапазону з обох сторін: є докази того, що залишки не відповідають нормальному розподілу.

(c) Прогнозування цін на акції зображено на рисунку 2. Фактичні ціни показано праворуч за допомогою порожніх кіл, щоб їх можна було порівняти з прогнозом регресії. Короткі сегменти (вгору та вниз) представляють стандартні помилки індивідуального прогнозу. Прогнози надзвичайно точні, і всі фактичні значення знаходяться в межах $\pm\hat{\sigma}\sqrt{1 + q(x)}$.

Неправильне тлумачення коефіцієнта детермінації

Зазвичай R^2, обчислене за формулою (5) неправильно тлумачиться як частка дисперсії, пояснена незалежною змінною/предиктором. Ця інтерпретація підтверджується розкладом залишкової суми квадратів (RSS):

$$\sum_{i=1}^{n}(Y_i - \overline{Y})^2 = \sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2 + \sum_{i=1}^{n}(\widehat{Y}_i - \overline{Y})^2, \quad (6)$$

де $\hat{Y} = \hat{\alpha} + \hat{\beta}x_i$ є прогнозованими значеннями. Виразивши $Y_i - \bar{Y} = (Y_i - \hat{Y}_i) + (\hat{Y}_i - \bar{Y})$, очевидно, що наведена вище рівність виконується тоді і тільки тоді, коли $\sum(Y_i - \hat{Y}_i)(\hat{Y}_i - \bar{Y}) = 0$. Щоб довести, що ця сума дорівнює нулю, пригадайте перше нормальне рівняння $\frac{\partial S}{\partial \beta} = 2\sum(Y_i - \hat{Y})x_i = 0$. Дійсно, оскільки $\sum(Y_i - \hat{Y}) = 0$, нормальне рівняння можна еквівалентно записати як $\sum(Y_i - \hat{Y})(x_i - \bar{x}) = 0$ Але оскільки $\hat{\alpha} = \bar{Y} - \hat{\beta}\bar{x}$, ми маємо $Y_i - \bar{Y} = \bar{Y} - \hat{\beta}\bar{x} + \hat{\beta}x_i - \bar{Y} = \hat{\beta}(x_i - \bar{x})$, тому

$$\sum(Y_i - \widehat{Y})(\widehat{Y}_i - \overline{Y}) = \widehat{\beta}\sum(Y_i - \widehat{Y})(x_i - \overline{x}) = 0.$$

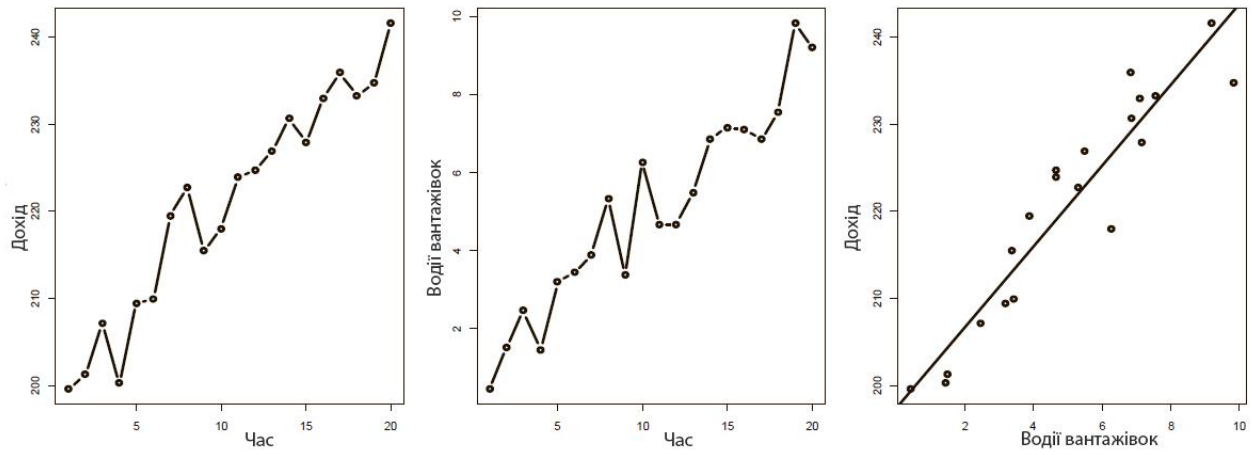
Розгортку RSS (6) можна розглядати як розгордку дисперсії, поділивши обидві сторони на *n*:

$$\frac{1}{n}\sum_{i=1}^{n}(Y_i - \overline{Y})^2 = \frac{1}{n}\sum_{i=1}^{n}(\widehat{Y}_i - \overline{Y})^2 + \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2,$$

яке можна розглядати як емпіричну версію розгордки дисперсії у двовимірному нормальному розподілі: ліва частина є оцінкою дисперсії Y , перший член у правій частині є оцінкою поясненої дисперсії, а другий член – це дисперсія непоясненої/залишкової дисперсії. Представляючи

$$R^2 = 1 - \frac{\sum_{i=1}^{n} r_i^2/n}{\sum_{i=1}^{n}(Y_i - \overline{Y})^2/n},$$

можна сказати, що чисельник оцінює непояснену дисперсію, знаменник оцінює дисперсію Y і, отже, R^2 оцінює пропорцію дисперсії Y, поясненої через x



Часовий ряд доходів компанії та кількості водіїв вантажівок. Як випливає з регресії, близько 90% доходу можна пояснити кількістю водіїв вантажівок.

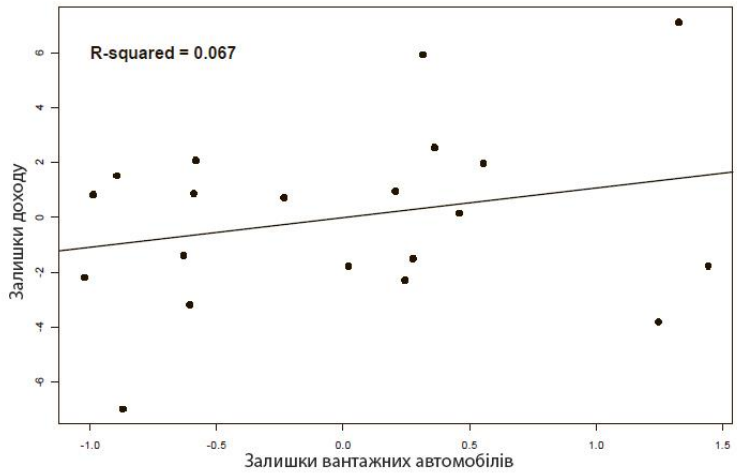
Проблема з наведеною вище аргументацією полягає в тому, що Y_i не однаково розподілені відповідно до лінійної моделі (0): Y_i мають різні середні, $E(Y_i) = \alpha + \beta x_i \neq \text{const}$. Існує фундаментальна різниця між регресією у двовимірному нормальному розподілі, де Y_i і X_i є iid, та лінійною моделлю (також називається регресією), де x_i є фіксованими числами. Ігнорування цієї різниці призводить до поширених помилок, таких як розширення наведеної вище теореми про статистичні властивості оцінки OLS на випадок, коли незалежна змінна x є випадковою і може спостерігатися разом із Y, або неправильним тлумаченням коефіцієнта детермінації як частки поясненої дисперсії.

Невиправдано високі значення R^2 можна очікувати, коли залежна змінна є часовим рядом, що призводить до помилкової регресії. Якщо значення Y_i збільшуються з часом, не є оцінкою дисперсії Y, а $\bar{Y}$ не є оцінкою середнього $\{Y_i, i = 1,2, \dots, n\}$. Якщо Y_i не є iid, стає великим порівняно з і означає, що R^2 близьке до 1. Іноді інтерпретація R^2 як частки поясненої дисперсії призводить до парадоксів, таких які еаведено в наступному прикладі.

Приклад. На вище наведеному малюнку зображено часовий ряд доходів компанії та кількість водіїв вантажівок. Регресуйте дохід від водіїв вантажівок і обчисліть коефіцієнт детермінації, R^2. Поясніть, чому R^2 не можна інтерпретувати як частку дисперсії доходу, поясненого водіями вантажівок, і запропонуйте альтернативний, більш обґрунтований коефіцієнт детермінації.

Розв'язання. Код R truckR створює наведену вище схему. Дотримуючись традиційної інтерпретації R^2, майже 90% коливань у доходах компанії пояснюється кількістю водіїв вантажівок — така регресія називається помилковою регресією. Це означає, що для покращення бізнесу генеральний директор повинен найняти більше водіїв вантажівок, що є

безглуздим рішенням. Дохід не є iid, тому двовимірний нормальний розподіл квадратичного коефіцієнта кореляції не застосовується. Щоб знайти надійний коефіцієнт кореляції, потрібно видалити тенденцію з обох часових рядів і співвіднести залишки (job=2). Дивіться наведену нижще схему: коефіцієнт кореляції падає до 0.067.



Кореляція між залишками. Хоча існує позитивна кореляція між доходом і кількістю водіїв вантажівок, вона незначна.

Розділ 3. Модедьний приклад

3.1. Постановка задачі

За останні 25 років відбувся значний науково-технічний прогрес. Людство навчилося конструювати роботів, які прибирають, миють, розмовляють, допомагають у будь-якій сфері людської діяльності. Сьогодні вже нікого не здивуєш туристичним польотом у космос чи обльотом міжконтинентальної ракети навколо земної кулі. Разом із стрімким розвитком техніки люди продовжують помирати від різних хвороб. Другою причиною смерті в світі після раку є смерть від інсульту. Згідно зі статистикою Всесвітньої організації охорони здоров'я, 11% усіх смертей відбувається внаслідок крововиливу в мозок - інсульту. Очевидно, що організм людини на клітинному рівні малодосліджений. Тому варто застосувати інтелектуальний аналіз даних, щоб встановити зв’язок між фізичними характеристиками пацієнтів, шкідливими звичками, віком тощо та випадками інсульту.

Основна мета роботи – знайти відповіді на такі запитання:

1. Чи впливає куріння на ймовірність інсульту?
2. Чи впливає гіпертонія на ймовірність інсульту?
3. Чи впливає вік на ймовірність інсульту?
4. Чи існує лінійна залежність між індексом маси тіла та середнім рівнем глюкози в організмі, віком пацієнта?

Щоб відповісти на ці запитання, ми розглянули моделі звичайної лінійної регресії та модель логістичної регресії. Їхня відмінність полягає в тому, що залежна змінна для лінійної регресії має бути числового типу, а для логістичної регресії – факторного. Отже, логістична регресія допоможе нам класифікувати пацієнтів за ймовірністю на дві групи (0 – НІ, 1 – ТАК). Окрім роботи над чотирма основними питаннями дослідження, нас також цікавить описова статистика, робота з відсутніми значеннями та статистичні тести. Інтелектуально встановивши фактори, що впливають на зростання ймовірності інсульту, кожен може зробити для себе відповідні висновки.

Набір даних

Дані були отримані з платформи Kaggle за наступним посиланням https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset. Дані складаються з 12 стовпців з 5110 записами. Інформація про атрибути:

1) id: унікальний ідентифікатор;
2) gender (стать): «Чоловік», «Жінка» або «Інше»
3) age (вік): вік пацієнта

4) hypertension (гіпертонія): 0 - якщо у пацієнта немає гіпертонії, 1 - якщо у пацієнта гіпертонія

5) heart_disease : 0 - якщо у пацієнта немає захворювань серця, 1 - якщо у пацієнта є захворювання серця

6) ever_married (одружений чи ні): "Ні" або "Так"

7) work_type (типроботи) : "діти", " державна ", " ніколи не працював ", "приватна"

8) Residence_type : "Сільський" або "Міський"

9) avg_glucose_level (середній рівень глюкози в крові)

10) bmi (індекс маси тіла)

11) smoking_status : "раніше курив", "ніколи не курив", "курить " або "Невідомо"

12) stroke (інсульт): 1 - якщо у пацієнта був інсульт, або 0 - якщо ні.

Для візуалізації світової карти рівня смертності від інсульту на 100 000 населення ми використовуємо ще один набір даних із https://www.worldlifeexpectancy.com/cause-of-death/stroke/by-country/.

Результати

Спочатку розглянемо вхідні дані ближче. Набір даних містить числові та факторні змінні. Наприклад, змінна gender є фактором типу (чоловічий, жіночий, інший).

На рисунку 3.1 показано індекс маси тіла за віком. Як бачимо, після 40 років людина з будь-яким індексом маси тіла може отримати інсульт. Отже, ймовірність появи інсульту зростає після 40 років для кожного. Ще один цікавий момент полягає в тому, що на рис. 3 індекс маси тіла не пов'язаний з інсультом. Пацієнти з різним індексом маси тіла можуть уникнути удару.

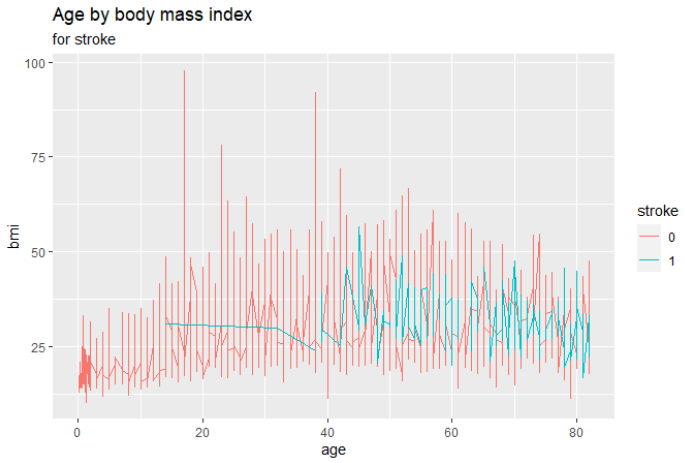


Рис.3.1.Індекс маси тіла за віком

Корелограма для наших даних показана на рисунку3.2. Як ми бачимо, атрибути (стовпці) у початковому наборі даних не корелюють. Кореляція

вважається слабкою, якщо коефіцієнт кореляції знаходиться в діапазоні від -0,3 до 0,3.

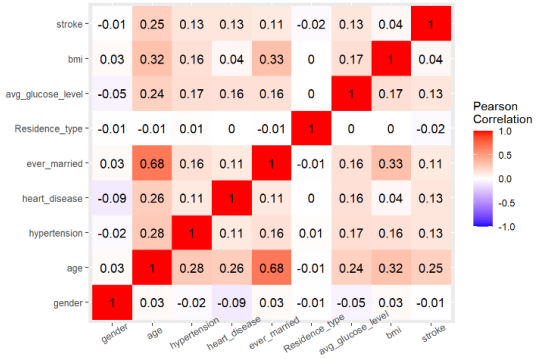


Рис.3.2. Корелограма

Тепер побудуємо модель лінійної регресії, щоб з'ясувати, чи існує лінійна залежність між індексом маси тіла та середнім рівнем глюкози. Рисунок 3.3 показує цю взаємозалежність. Індекс маси тіла тут є залежною змінною, а середній рівень глюкози є незалежною змінною.

Рівняння лінійної регресії має такий вигляд:

$$bmi = 25.63 + 0.03 * avg\_glucose\_level$$

Тепер подивимося, як вік впливає на середній рівень глюкози. Для цього ми побудуємо іншу модель лінійної регресії та побудуємо графік – рис. 3.4 . Як видно з рис. 3.4, із збільшенням кількості років пацієнта середній рівень глюкози також зростає. Тобто чим старше ми стаємо, тим менше цукру нам потрібно вживати. Рівняння лінійної регресії для цієї взаємозалежності має вигляд:

$$\boldsymbol{average\ glucose\ level = 85.53 + 0.47 * age}$$

Слід зазначити, що обидві побудовані моделі не є репрезентативними. Тобто існують інші змінні, які впливають на змінну відповіді. Такий висновок ми отримали після аналізу значень коефіцієнтів R^2. Модель вважається прийнятною, якщо R^2вона перевищує 80%. У нашому випадку R^2це близько 10% в обох випадках.

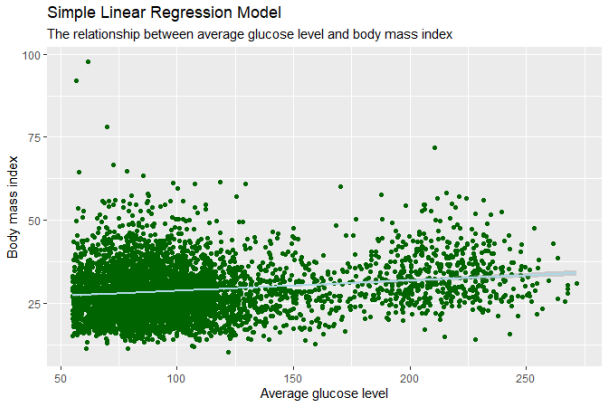


Рис.3.3.Модель лінійної регресії для змінної відповіді індексу маси тіла

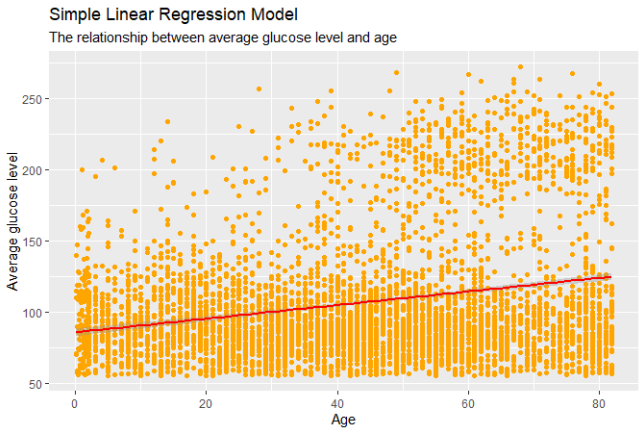


Рис.3.4.Модель лінійної регресії для змінної реакції середнього рівня глюкози

Лінійна регресія не може бути використана для визначення ймовірності того чи буде пацієнт мати інсульт. Тобто відповідна змінна "stroke" у нашому випадку може приймати лише 2 можливих значення. Ось де на допомогу приходить логістична регресія. Логістична регресія дозволяє отримати відповідь у вигляді ймовірності від 0 до 1.

Таблиця 3.1. показує результати моделі логістичної регресії. Ми не перераховуємо всі незалежні змінні в таблиці, щоб не обтяжувати звіт. Змінні, не зазначені в таблиці, є статистично незначущими. Тобто вони не впливають на змінну stroke.

Таблиця 3.1. Результати регресії

	Оцінка	Pr(>|t|)
(Intercept)	-5,922908	2.14e-13 ***
age(вік)	0,072741	< 2e-16 ***
hypertension1(гіпертонія)	0,563302	0,00348 **
heart_disease1 (серце-ва_хвороба)	0,342615	0,13270
ever_marriedYes	-0,233803	0,37770
Residence_typeUrban	-0,102680	0,53266
avg_glucose_level	0,002709	0,06287.
` work_type_Self -employed`	-1,438607	0,09906
`smoking_status_never smoked	-0,244233	0,20289
smoking_status_smokes	-0,074173	0,77644

Тому змінні age (вік) та hypertension (гіпертонія) є статистично значимим для ймовірністі отримати інсульт.

Висновки

В даній магістерській роботі було розглянуто статистичні підходи, які застосовуються у біоінформатиці.

В роботі досліджено модельний приклад про пацієнтів, де встановленно фактори впливу на ймовірність одержання інсульту пацієнтом. Отримано аналіз даних про встановлення зв'язку між фізичними характеристиками пацієнта, його шкідливими звичками, способом життя та ймовірністю виникнення інсульту. Для встановлення зв’язку між числовими даними були побудовані моделі лінійної регресії. Оскільки змінна відповіді «інсульт» має факторний тип, була побудована модель логістичної регресії для прогнозування ймовірності інсульту.

Аналіз зроблено за допомогою середовища R.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Baldi P, Brunak S. Bioinformatics. The Machine Learning Approach. MIT Press, 2001.

2. Huang, Xiao & Cao, Tianyu & Chen, Liangziqian & Li, Junpei & Tan, Ziheng & Xu, Benjamin & Xu, Richard & Song, Yun & Zhou, Ziyi & Wang, Zhuo & Wei, Yaping & Zhang, Yan & Li, Jianping & Huo, Yong & Qin, Xianhui & Wu, Yanqing & Wang, Xiaobin & Wang, Hong & Cheng, Xiaoshu & Liu, Lishun. (2022). Novel Insights on Establishing Machine Learning-Based Stroke Prediction Models Among Hypertensive Adults. Frontiers in Cardiovascular Medicine. 9. 901240. 10.3389/fcvm.2022.901240

3. Neil C. Jones, Pavel A. PevznerAnIntroduction to Bioinformatics Algorithms. Cambridge, Massachusetts: London.- 2004. – 436 p.

4. Saleh, Hager & F Abd-el Ghany, Sara & Younis, Eman & Omran, Nahla & Ali, Abdelmgeid. (2022). Stroke Prediction using Distributed Machine Learning Based on Apache Spark. International Journal of Advanced Science and Technology. Vol. 28, No. 15

Додаток

```
library(readr)
library(missMethods)
library(tidyverse)
library(broom)
library(mice)
library(RCurl)
library(ggplot2)
library(raster)
library(ggthemes)
library(scales)
library(conflicted)
library(plyr)
library(Amelia)
library(usmap)

library(plotly)
library(tools)
library(corrplot)

library(tidyverse)
library(dplyr)
library(reshape)
library(ggplot2)
library(corrgram)

df <- read_csv("healthcare-dataset-stroke-data.csv")
View(df)

options(warn=-1)
df$age <- as.numeric(df$age)
df$gender <- as.factor(df$gender)
df$smoking_status <- as.factor(df$smoking_status)
df$stroke <- as.factor(df$stroke)
df$avg_glucose_level <- as.numeric(df$avg_glucose_level)
df$bmi <- as.numeric(df$bmi)

str(df)

###Ділянка Корелограма
# Residence_type: 0 - міський, 1 - сільський
stroke1$Residence_type[stroke1$Residence_type == "Urban"] <- 0
stroke1$Residence_type[stroke1$Residence_type == "Rural"] <- 1

# ever_married: 0 - ні,  1 - так
stroke1$ever_married[stroke1$ever_married == "Yes"] <- 1
stroke1$ever_married[stroke1$ever_married == "No"] <- 0

# gender: 0 - чоловіча, 1 - жіноча
stroke1$gender[stroke1$gender == "Male"] <- 0
stroke1$gender[stroke1$gender == "Female"] <- 1

suppressWarnings(stroke1$Residence_type <- as.numeric(as.character(stroke1$Residence_type)))
```

```
suppressWarnings(stroke1$ever_married <- as.numeric(as.character(stroke1$ever_married)))
suppressWarnings(stroke1$gender <- as.numeric(as.character(stroke1$gender)))

# кількісні змінні: карта кореляції
stroke.quant = subset(stroke1, select = -c(work_type, smoking_status))
stroke.cor = round(cor(stroke.quant),2)
ggplot(data = reshape2::melt(stroke.cor),aes(x=Var1, y=Var2, fill=value)) + geom_tile() +
scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit = c(-1,1), space = "Lab",
name="Pearson\nCorrelation") + geom_text(aes(Var2, Var1, label = value), color = "black", size = 4) +
theme(axis.text.x = element_text(angle = 30))

ggplot(data = df, aes(y = gender, legend = T)) +
  geom_bar(fill = rainbow(3)) + labs(title = "Gender barplot")

ggplot(data = df, aes(x = avg_glucose_level, y = bmi, color = gender)) +
  geom_line() + labs(title = "Average glucose level in blood by body mass index",
                subtitle = "for gender")

ggplot(data = df, aes(x = age, y = bmi, color = stroke)) +
  geom_line() + labs(title = "Age by body mass index",
                subtitle = "for stroke")

ggplot(data = df, aes(y = smoking_status, legend = T)) +
  geom_bar(fill = rainbow(4)) + labs(title = "Smoking status barplot barplot")

ggplot(data = df, aes(y = stroke, legend = T)) +
  geom_bar(fill = rainbow(2)) + labs(title = "Smoking status barplot barplot")

###Як бачимо, кількість вбивств значно вища в одному штаті порівняно з іншими
missmap(df)
colSums(is.na(df))

# дослідження колонки gender
as.data.frame(table(df$gender))

#замінаіндексу маси тіла на медіану.
bmi.median = median(df$bmi, na.rm = TRUE)
df$bmi <- df$bmi %>% replace_na(bmi.median)
missmap(df)
colSums(is.na(df))

###Перевірка відсутніх даних
is.na(df)
which(is.na(df))
df%>%
  dplyr::filter(!complete.cases(.))%>%
  view()
md.pattern(df)

missmap(df)
sum(is.na(mydata))

df$corr <- cor(df$avg_glucose_level, df$bmi)
```

```
###Отже, вхідний набір даних не містить значень NA
# Rename our dataset
mydata <- df
### Generating a function to generate random "NA" except the first 2 columns (Rank,Region) na.gen =
function(mydata,n ){
na.gen = function(mydata,n ){
 i=1
 while( i < n+1){
  idx1 = sample(nrow(mydata), 1)
  idx2 = sample(3:ncol(mydata), 1)
  mydata[idx1,idx2] = NA
  i = i+1
 }
return(mydata)
}



#2.2додавання випадкової NA до даних
mydata = na.gen(mydata, 100)


#2.3 повторна перевірка відсутніх даних
sum(is.na(mydata))
view(colSums(is.na(mydata)))
#colSums(is.na(mydata))[1,]

#2.4 список рядків з відсутніми даними

missingdata = mydata[!complete.cases(mydata),]

sum(is.na(missingdata))
view(missingdata)


#Handling the missing data with Amelia
#IMPUTATIONS USING AMELIA
summary(mydata)
mydata$corr <- NULL
mydata <- as.data.frame(mydata)
imp_amelia = amelia(x = mydata, m = 30,idvars= c("state"))

amelia

summary(imp_amelia)


### leaner8

lm.amelia = lapply(imp_amelia$imputations, function(i)lm(Murder ~       unemployment,data = i))
### зведення всіх коефіцієнтів
```

```
coefs.amelia = lm.amelia %>%
sapply(coef) %>%
t()
my_SE_extractor = function(model){summary(model)$coef[,"Std. Error"]}
ses.amelia = lm.amelia %>%
sapply(my_SE_extractor) %>%
t()


###тепер ми готові до використання mi.meld
mi.meld(coefs.amelia, ses.amelia)

###перевірка відсутніх даних після імпутації
missmap(mydata)
sum(is.na(mydata))
table(sum(is.na(mydata)))
missmap(imp_amelia)
sum(is.na(imp_amelia))
sum(is.na(imp_amelia))

#перевірка гіпотези

hypo_model1 = lm(df$bmi ~ df$avg_glucose_level,data = df)
summary(hypo_model1)

hypo_model2 = lm(df$avg_glucose_level ~ df$age,data = df)
summary(hypo_model2)
### Оскільки фактор групування повинен мати рівно 2 рівні, щоб t.test запускався для
наведеноївище моделі
### ми використовуємо тест aov, щоб знайти суму квадратів і ступенів свободи та довірчий
інтервал
confint(aov(Murder ~ unemployment,data = imp_amelia$imputations$imp15))

View(df)
hypo_model1 = ggplot(df,
             aes(x=avg_glucose_level,y = bmi)) + geom_point(color= "dark
green")+(aes(x=avg_glucose_level,y = bmi)) +geom_smooth(method="lm", color = "light blue") +
labs(title = "Simple Linear Regression Model" ,
          subtitle = "The relationship between average glucose level and body mass index",
          x = "Average glucose level",
          y = "Body mass index") + theme(axis.title = element_text())
hypo_model1
summary(hypo_model1)

hypo_model2 = ggplot(df,
             aes(x=age,y = avg_glucose_level)) + geom_point(color= "orange")+(aes(x=age,y =
avg_glucose_level)) +geom_smooth(method="lm", color = "red") + labs(title = "Simple Linear Regression
Model" ,

subtitle = "The relationship between average glucose level and age",
x = "Age",
y = "Average glucose level") + theme(axis.title = element_text())
```

```
hypo_model2
summary(hypo_model1)

### Розділяємо дані на навчальні та тестові
sample <- sample(c(TRUE, FALSE), nrow(df), replace = T, prob = c(0.6,0.4))
train <- df[sample, ]
test <- df[!sample, ]

### Логістична регресійна модель
logmodel1 <- glm(stroke ~ age +
hypertension+heart_disease+ever_married+work_type+Residence_type+avg_glucose_level+bmi+smoki
ng_status, family = "binomial", data = train)
summary(logmodel1)

logmodel2 <- glm(stroke ~ hypertension+bmi+avg_glucose_level+smoking_status, family = "binomial",
data = train)

### Результат регресії
summary(logmodel2)


library(tidyverse)
library(maps)
library(highcharter)
library(renv)

stroke <- read_delim("Stroke.csv", ";", escape_double = FALSE,
                     col_names = FALSE, col_types = cols(X4 = col_skip()),
                     locale = locale(decimal_mark = ","),
                     trim_ws = TRUE)
View(stroke)
stroke <- stroke[,1:3]
dat <- iso3166
head(dat)
dat <- dplyr::rename(dat, "iso-a3" = a3)
head(dat)

countries_visited <- c(stroke$X2)
dat$visited <- ifelse(dat$ISOname %in% countries_visited, 1, 0)
View(dat)

hcmap(
  map = "custom/world-highres3", # high resolution world map
  data = dat, # name of dataset
  joinBy = "iso-a3",
  value = "visited",
  showInLegend = FALSE, # hide legend
  nullColor = "#DADADA",
  download_map_data = TRUE
) %>%
  hc_mapNavigation(enabled = FALSE) %>%
  hc_legend("none") %>%
  hc_title(text = "World map") # title
```

```
dat <- subset(dat, dat$visited == 1)
sort(dat$ISOname)
```