Міністерство освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Прикладна біоінформатика

Розрахунково-графічна робота

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня магістра
за освітньою програмою «Біотехнології»
спеціальності 162 Біотехнології та біоінженерія

Укладач: С.В. Горобець

Електронне мережеве навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2024

УДК 004.43 (004)
П75

Укладач: Горобець Світлана Василівна, доктор техн. наук, проф. кафедри біоенергетики, біоінформатики та екобіотехнології, факультету біотехнології і біотехніки, КПІ ім. Ігоря Сікорського

Рецензент Шибецький Владислав Юрійович. канд. техн. наук, доцент кафедри біотехніки та інженерії, факультету біотехнології і біотехніки, КПІ ім. Ігоря Сікорського

Відповідальний редактор Козар Марина Юріївна, канд. техн. наук, доцент кафедри біоенергетики, біоінформатики та екобіотехнології, факультету біотехнології і біотехніки, КПІ ім. Ігоря Сікорського

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 8 від 20.06.2024 р*)*
за поданням вченої ради факультету біотехнологій та біотехніки
(протокол № 10 від 27.05.2024 р.)

П75 **Прикладна біоінформатика.** Розрахунково-графічна робота [Електронний ресурс]: навч. посіб. для здобувачів ступеня магістра за освіт. програмою «Біотехнології» спец. 162 Біотехнології та біоінженерія / КПІ ім. Ігоря Сікорського ; уклад.: С. В. Горобець. – Електрон. текст. дані (1 файл). – Київ : КПІ ім. Ігоря Сікорського, 2024. – 23 с.

В навчальному посібнику викладено вимоги до підготовки та оформлення розрахунково-графічної роботи з дисципліни «Прикладна біоінформатика». Подано матеріал, який відображає особливості роботи з бібліотеками мови програмування Python, які призначені для задач машинного навчання. Розрахунково-графічну роботу присвячено методу дерев рішень (decision trees) – одного з найбільш популярних методів розв'язання задач класифікації, що широко використовується для біомедичних застосувань.

УДК 004.43 (004)

Реєстр. № НП 23/24-640 Обсяг 0.9 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Берестейський, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів
і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

ЗМІСТ

ВСТУП

Провідні фармацевтичні компанії світу для створення передової терапії та розвитку персоналізованої медицини використовують машинне навчання та штучний інтелект для пошуку нових молекул та їх комбінацій: Roche (53bn), Pfizer (49.7bn), Johnson & Johnson (42.2bn), Merck (41.8bn), Novartis (37.7bn), AbbVie (33.3bn), Takeda (31.1bn), Bristol–Myers Squibb (26.1bn).

Наступні бібліотеки мови python використовуються для інженерних та наукових досліджень, в тому числі для задач регресійного аналізу та машинного навчання: numpy, scipy, scikit-learn (sklearn), matplotlib, pandas.

Пакет numpy (numeric python) – це бібліотека на мові python, що допомагає підтримувати більшість різноманітних масивів і матриць, разом з великою бібліотекою високорівневих (і дуже швидких) математичних функцій для операцій з цими масивами. Numpy – досить низькорівневий інструмент, схожий на MATLAB.

Пакет pandas надає багаті функціональні можливості для аналізу часових рядів, аналізу даних, зручну для використання статистику, методи злиття і об'єднання даних, а також безліч інших можливостей. Бібліотека стала дуже популярною в останні роки в фінансових додатках. Бібліотека pandas спрощує життя аналітикам: де раніше використовувалося 10 рядків коду тепер вистачить одного.

Пакет scipy (scientific python) – це відкрита бібліотека високоякісних інструментів в науці і для інженерних розробок для мови програмування python, містить модулі для оптимізації, інтегрування, обробки сигналів, обробки зображень, генетичних алгоритмів, розв'язування звичайних диференціальних рівнянь тощо. Numpy і scipy – це бібліотеки python, які використовуються для математичного та числового аналізу. Якщо ви імпортуєте scipy, то numpy окремо імпортувати не потрібно.

Пакет scikit-learn (sklearn) – це бібліотека машинного навчання з відкритим вихідним кодом. Пакет sklearn також надає різні інструменти для

побудови моделей, попередньої обробки даних, вибору та оцінки якості моделі тощо. Пакет sklearn побудований на основі пакетів numpy, scipy та matplotlib.

Останнім часом синтез металевих наночастинок (MtNP) з використанням мікроорганізмів і рослин був визнаний ефективним і екологічно чистим методом зеленого синтезу, в якому мікроорганізми використовуються як нанофабрики отримання MtNP для біомедичних застосувань та в сільському господарстві.

Перевагами розвитку зелених нанотехнологій, що використовують біологічні шляхи для синтезу наноматеріалів, є [1]:

- Кількість енергії, що витрачається в зелених нанотехнологіях, значно нижча, ніж в інших технологіях.
- Мінімізація виробництва небезпечних речовин, так як при зеленому синтезі майже не утворюється токсичних хімічних речовин.
- Мінімізація відходів при виготовленні продуктів, що є більш ефективним.
- Скорочення використання добрив і пестицидів з використанням носіїв і сполук на основі наночастинок. При використанні пестицидів і добрив наночастинки використовуються як носії, які поступово вивільняють активні інгредієнти, щоб зменшити їх загальне споживання.
- Підвищення якості ґрунту та якості сільськогосподарської продукції, використовуючи добрива і пестициди на основі наночастинок, або стимулюючи схожість сільськогосподарського насіння та ріст рослин безпосередньо за допомогою наночастинок, оскільки магнітні наночастинки (МНЧ) здатні підвищити схожість сільськогосподарського насіння та урожайність рослин.

Зелений синтез MtNP за допомогою мікроорганізмів має низку переваг порівняно зі звичайними фізико-хімічними методами. Зокрема, він пропонує швидкий, економічно ефективний, нетоксичний та екологічно чистий метод синтезу MtNP з широким діапазоном розмірів, форм, складу та фізико-хімічних властивостей. Однак основні недоліки синтезу MtNP на основі

мікроорганізмів включають складні етапи, такі як підбір мікроорганізмів, здатних до синтезу MtNP, культивування, виділення MtNP та зберігання. Крім того, відновлення MtNP, отриманих цим методом, вимагає подальшої обробки.

Важливим напрямком біомедичного застосування металевих наночастинок є їх отримання з магнітокерованими властивостями. Сучасні підходи до намагнічування клітин за допомогою магнітних наночастинок (МНЧ) вимагають їх поглинання клітинами або адсорбції за допомогою маніпуляцій з клітинами *in vitro*. Відносно новою галуззю досліджень є «магнітогенетика», яка зосереджується на виробництві та накопиченні магнітного матеріалу *in vivo*. Мікроорганізми з природними магнітокерованими властивостями синтезують біогенні магнітні наночастинки (БМН), генетичний механізм синтезу яких докладно досліджено у магнітотаксисних бактерій (МТБ). МТБ утворюють внутрішньоклітинні ліпосоми, які називаються магнітосомами, наповнені кристалічними МНЧ – магнетитом або грейгітом. МТБ також демонструють ланцюгоподібне розташування магнітосом. В останні роки дослідники з'ясували функцію та структуру багатьох білків МТБ. Крім того, дослідження білкової інженерії таких білків МТБ, що відповідають за властивості БМН (намагніченість, розмір, форма тощо), є перспективною галуззю досліджень.

Магнітні наночастинки стали важливими після успішного використання мічених ними антитіл для розділення клітин. Відтоді розділення клітин за допомогою МНЧ часто використовується для виявлення та ізоляції клітин із складних сумішей [2]. Використання МНЧ також представляє значний інтерес як контрастний агент для магнітно-резонансної томографії (МРТ) і магнітно-порошкової візуалізації [3], для цільової доставки ліків [4], регенеративної медицини на основі стовбурових клітин [5], лікування гіпертермією [6], а також магнітомеханічної регуляції метаболізму клітин [7].

Для синтезу БМН широко використовуються магнітотаксисні бактерії, які накопичують магнітні мінерали приблизно до 3% їхньої сухої ваги, їхні магнітосоми демонструють однодоменну магнітну поведінку, мають розміри

від 35 до 120 нм і є золотим стандартом МНЧ [8-10]. Генна інженерія немагнітних господарів, таких як *R. rubrum* з 29 генами магнітосом з *M. gryphiswaldense*, дозволила отримати магнітокеровану клітину. Це дослідження також дало найбільші рекомбінантно отримані БМН (~24 нм) у спочатку немагнітокерованої клітині. Однак перенесення великого пулу білків МТБ до немагнітокерованих хазяїв не завжди дає можливість отримати МНЧ. Крім того, експресія білків МТБ є складною у звичайних промислових господарях, таких як, наприклад, бактерія *E. coli* [11].

Значне різноманіття МТБ, а відповідно і властивостей БМН, які вони синтезують, вимагає величезної кількості експериментальних досліджень цих властивостей, що привело до розвитку передових напрямків, що стрімко розвиваються: Machine Learning (машинне навчання), Data Mining (аналіз даних) та Big Data (робота з великими масивами даних). Це одна з причин, чому з осені 2021 й дотепер python посідає чільне місце в рейтингу найбільших компаній світу GitHub Inc., TIOBE (The Importance of Being Earnest) та ін. Для задач машинного навчання використовуються наступні бібліотеки мови python numpy, sklearn, scipy, matplotlib, pandas тощо [1-4].

1. ТЕОРЕТИЧНА ЧАСТИНА

1.1. Метод дерев рішень – Data Mining

Метод дерев рішень (decision trees) є одним із найбільш популярних методів розв’язання задач класифікації й прогнозування. Іноді цей метод Data Mining також називають деревами вирішальних правил, деревами класифікації і регресії [5].

Як видно з останньої назви, за допомогою даного методу розв’язуються задачі класифікації й прогнозування:

- Якщо залежна, тобто цільова змінна приймає дискретні значення, то за допомогою методу дерева рішень розв’язується задача кластерізації (класифікації).

- Якщо ж залежна змінна приймає неперервні значення, то дерево рішень встановлює залежність цієї змінної від незалежних змінних, тобто розв’язує задачу чисельного прогнозування.

Дерево рішень – це блок-схема, яка може допомогти приймати рішення на основі вихідних даних або на основі попереднього досвіду [5].

У найбільш простому вигляді дерево рішень – це спосіб формування правил в ієрархічній, послідовній структурі. Основа такої структури – відповіді «так» або «ні» на низку питань. Тобто потрібно відповісти на низку питань, які є у вузлах дерева, починаючи з його кореня. Перший вузол дерева є вузлом перевірки, тобто умовою. При позитивній відповіді на запитання здійснюється перехід до лівої частини дерева, що називається лівою гілкою, при негативному – до правої частини дерева, що називається правою гілкою. Отже, внутрішній вузол дерева також є вузлом перевірки певної умови. Далі йде наступне питання і т.д., поки не буде досягнутий кінцевий вузол дерева, що є вузлом розв’язку. Вузли дерева, за якими не слідує гілка називаються листями дерева.

Характерна риса розщеплення у вузлах дерева: кожний запис (іншими словами кожен елемент вибірки даних) використовує унікальний шлях від кореня дерева тільки до одного вузла-розв’язку. Об’єднана інформація про розщеплення у вузлі називається критерієм розщеплення (splitting criterion). Кожна гілка дерева, що йде від внутрішнього вузла, характеризується критерієм розщеплення. Процес створення дерева відбувається зверху вниз, тобто є спадаючим. У ході процесу побудови дерева алгоритм повинен знайти такий критерій розщеплення, який іноді називається критерієм розбивки, щоб розбити множину на підмножини, які б асоціювалися з даним вузлом перевірки. Кожний вузол перевірки повинен бути позначений певним критерієм перевірки. Існує правило вибору критерію: він повинен розбивати вихідну множину даних таким чином, щоб об’єкти підмножин, що одержуються в результаті цієї розбивки, були представниками одного класу або ж були максимально наближені до такої розбивки. Це означає, що кількість

об’єктів з інших класів, так званих «домішок», у кожному класі повинна прямувати до мінімуму. Існують різні критерії розщеплення. Найбільш відомі з них – міра ентропії, індекс Gini тощо. Індекс Gini використовує наступну формулу: $Gini = 1 - (x/n)^2 - (y/n)^2$, де x – кількість позитивних відповідей («так»), y – кількість негативних відповідей («ні»), n - кількість зразків (елементів вибірки).

Внутрішні вузли дерева є атрибутами моделі. Як вже зазначалося, ці атрибути називають прогнозуючими, або атрибутами розщеплення (splitting attribute). Кінцеві вузли дерева, або листки, іменуються мітками класу, що є значеннями залежної категоріальної змінної.

У результаті проходження від кореня дерева (іноді називається кореневою вершиною) до його вершини розв’язується задача класифікації, тобто вихідні дані розділяються на декілька класів.

Мета виконання розрахунково-графічної роботи **–** використання пакетів мови програмування Python для задач кластерізації.

1.2. Класи і методи реалізації алгоритму дерева рішень з використанням модуля tree пакету sklearn

1.2.1. Клас DecisionTreeClassifier()

sklearn.tree.DecisionTreeClassifier (criterion='gini', splitter='best', max_depth= None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_ decrease=0.0, class_weight=None, ccp_alpha=0.0) [2]

Параметри класу DecisionTreeClassifier:

- criterion ({“gini”, “entropy”, “log_loss”}, default=”gini”) – функція вимірювання якості розщеплення. Підтримувані критерії: “gini”, “log_loss” та “entropy”.

- splitter ({“best”, “random”}, default=”best”) – стратегія, яка використовується для вибору розщеплення на кожному вузлі. Підтримувані стратегії: “best” для вибору найкращого розщеплення та “random” для вибору найкращого випадкового розщеплення.
- max_depth (int, default=None) – максимальна глибина дерева. Якщо None, то вузли розгортаються, доки всі листи не стануть чистими або поки всі листя не міститимуть мінімальну кількість зразків, min_samples_split.
- min_samples_split (int або float, default=2) – мінімальна кількість зразків, необхідних для розбиття внутрішнього вузла: якщо int, вважайте min_samples_split мінімальним числом; якщо float, min_samples_split – це ціле число від добутку min_samples_split * n_samples – мінімальна кількість зразків для кожного розщеплення.
- min_samples_leaf (int or float, default=1) – мінімальна кількість зразків, необхідна для листкового вузла. Точка розщеплення на будь-якій глибині розглядатиметься, лише якщо вона залишає принаймні min_samples_leaf навчальних зразків у кожній із лівої та правої гілок. Це може мати ефект згладжування моделі, особливо в регресії. Якщо int, то вважайте min_samples_leaf мінімальним числом. Якщо float, то ціле число від добутку min_samples_leaf * n_samples є мінімальною кількістю зразків для кожного вузла.
- min_weight_fraction_leaf (float, default=0.0) – мінімальна зважена частка загальної суми ваг (усіх вхідних вибірок), яка повинна бути на листовому вузлі. Зразки мають однакову вагу, якщо sample_weight не надано.
- max_features (int, float або {“auto”, “sqrt”, “log2”}, default=None) – кількість ознак, які слід враховувати під час пошуку найкращого розщеплення:

- якщо int, тоді враховуйте max_features ознак при кожному розщепленні.

- якщо float, то max(1, int(max_features * n_features_in_)) враховуються під час кожного розщеплення.
- якщо «sqrt», то max_features=sqrt(n_features).
- якщо «log2», то max_features=log2(n_features).
- якщо None, то max_features=n_features.

- random_state (int, RandomState instance або None, default=None) – контролює випадковість оцінювача. Ознаки завжди випадково переставляють під час кожного розщеплення, навіть якщо для розділювача встановлено значення = best". Коли max_features < n_features, алгоритм випадково вибере max_features для кожного розщеплення перед тим, як знайти найкраще з них. Але найкраще знайдене розщеплення може відрізнятися в різних запусках, навіть якщо max_features = n_features. Це має місце, якщо покращення критерію є ідентичним для кількох розщеплень і одне розщеплення має бути обрано випадковим чином. Щоб отримати детерміновану поведінку під час відладки, random_state має бути зафіксовано як ціле число.
- max_leaf_nodes (int, default=None) – побудова дерева з max_leaf_nodes у найкращий спосіб. Найкращі вузли визначаються як відносне зменшення домішок. Якщо None, то це означає необмежену кількість листових вузлів.
- ccp_alpha (non-negative float, default=0.0) – параметр складності, який використовується для скорочення мінімальної вартості та складності. Буде обрано піддерево з найбільшою вартістю складності, меншою за ccp_alpha. За замовчуванням обрізка не виконується.

Атрибути класу DecisionTreeClassifier:

- classes_ (ndarray of shape (n_classes,) або list of ndarray) – мітки класів (задача з одним виходом) або список масивів міток класів (задача з кількома виходами).
- feature_importances_ (ndarray of shape (n_features,)) – повертає важливість ознак.

- max_features_ (int) - повертає виявлене значення max_features.
- n_classes_ (int or list of int) – кількість класів (для задач з одним виходом) або список, що містить кількість класів для кожного виходу (для задач з кількома виходами).
- n_features_in_ (int) – кількість ознак, які видно під час побудови дерева.
- feature_names_in_ (ndarray of shape (n_features_in_,)) – назви ознак, які можна побачити під час підгонки. Визначається лише тоді, коли X має імена ознак, які є рядками.
- n_outputs_ (int) – кількість виходів під час підгонки.
- tree_ (Tree instance) – базовий об'єкт Tree. Будь ласка, зверніться до довідки (sklearn.tree._tree.Tree) щодо атрибутів об’єкта Tree та розуміння структури дерева рішень для базового використання цих атрибутів.

Примітка: пошук розщеплення не припиняється, доки не буде знайдено принаймні один дійсний розділ вузлів зразків, навіть якщо для цього потрібно ефективно перевірити більше, ніж max_features ознак.

1.2.2. Метод fit() модуля tree пакету sklearn для побудови дерева рішень

Метод fit() зазвичай приймає 2 параметри [2]:

Матрицю X (n_samples, n_features), в якій дані представлені у вигляді рядків, а ознаки представлені у вигляді стовпчиків.

Цільові значення y, які є дійсними числами для задач регресії, або цілими числами для класифікації (або будь-якого іншого дискретного набору значень). Для некерованих навчальних задач (коли навчальні дані відсутні) y не потрібно вказувати. Зазвичай y є одновимірним масивом, де i-ий запис відповідає цільовому i-ому рядку X. Обидва масиви X та y зазвичай є масивами пакету numpy або еквівалентними типами даних, подібними до масиву.

Метод fit() повертає дані в спеціальному форматі, який необхідний для передачі цих даних методу plot_tree(), який відображає дендрограму дерева рішень.

1.2.3. Метод plot_tree() модуля tree пакету sklearn для відображення дендрограми дерева рішень

sklearn.tree.plot_tree(decision_tree, *, max_depth=None, feature_names=None, class_names=None, label='all', filled=False, impurity=True, node_ids=False, proportion=False, rounded=False, precision=3, ax=None, fontsize=None) [2].

В методі plot_tree() будує дерево рішень. Налаштування візуалізації дерева здійснюється автоматично, аргументи figsize або dpi plt.figure використовуються для керування розміром візуалізації.

Параметри методу plot_tree():

decision_tree – класифікатор дерева рішень, тобто дерева рішень, яке буде побудовано.

max_depth, default=None – максимальна глибина побудови дерева. Якщо максимальна глибина не задана, то відображається все дерево.

feature_names, default=None – масив типу str, назви кожної з ознак. Якщо назви ознак не задані, використовуватимуться загальні назви (“x[0]”, “x[1]”, …).

class_names, default=None – масив str або True, назви кожного з цільових класів у порядку зростання числа. Цей параметр використвується лише для класифікації. Якщо цей параметр True, то відображається символьне представлення імені класу.

label{‘all’, ‘root’, ‘none’}, default=’all’ – чи показувати інформаційні мітки для даних тощо. Опції включають: «all» – для відображення в кожному вузлі, «root» для відображення лише в верхньому кореневому вузлі, «none» – задається для того, щоб не відображати інформаційні мітки на жодному вузлі.

filled, default=False – якщо встановлено значення True, вузли зафарбовуються для позначення класу для задач класифікації.

impurity, default=True – якщо встановлено True, будуть відображені домішки в кожному вузлі.

node_ids, default=False – якщо встановлено True, буде відображено ідентифікаційний номер на кожному вузлі.

proportion, default=False – якщо встановлено True, буде змінено відображення «значень» та/або «зразків» на пропорції та відсотки відповідно.

rounded, default=False – якщо встановлено True, блоки вузлів буде зображено з закругленими кутами, шрифти Helvetica будуть використовуватися замість Times-Roman.

precision, default=3 – int, кількість цифр точності для числа з плаваючою комою в значеннях домішок, порогових значеннях та значеннях атрибутів (тобто значень ознак, за якими відбувається класифікація в кожному вузлі) кожного вузла.

ax=None – вісь matplotlib, осі для побудови дерева, типово – ax=None. Якщо осі для побудови не задано, використовується поточна вісь, будь-який попередній вміст очищається.

fontsize, default=None – int, розмір шрифту тексту. Якщо розмір шрифту тексту не задано, визначається автоматично, щоб відповідати розміру рисунку дерева.

2. ПОСЛІДОВНІСТЬ ВИКОНАННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Для реалізації методу дерева рішень з використанням модуля tree пакету sklearn необхідно:

- Підібрати дані для задачі. Для задач кластеризації і подібних задач машинного навчання дані можна отримувати з однієї чи декількох баз даних або з наукових статей, при цьому в обох випадках з додаванням або без додавання своїх експериментальних або біоінформатичних досліджень.
- Реалізувати на мові програмування python алгоритм дерева рішень з використанням підібраних для задачі даних та з використанням описаних в розділі 1 класів та методів модуля tree пакету sklearn.

2.1.Формування вихідних даних та написання коду на мові програмування python для побудови дерева рішень

2.1.1.Створення файлу з даними на диску

Необхідно в директорії, в якому Ви будете писати код, створити файл з розширенням .txt або файл з розширенням .xlsx, наприклад, DecisionTree.txt або DecisionTree.xlsx.

Дані для навчання класифікатора сформуйте на основі даних роботи Peiyu Liu [6] (рис.1), в якій встановлено зв'язок між формою біогенних магнітних наночастинок і генами (білками) магнітосом МТБ (рис. 1).

На рис. 1 по вертикалі представлено гени (білки) або набори генів (білків): MACG – набір основних генів магнітосом у системі МТВ (тобто mamA, mamB, mamE, mamI, mamK, mamM, mamP і mamQ); набір генів MACGP, гени магнітосом, збережені у Pseudomonadota (тобто mamH, mamF, mamS і mamT), та наступні гени для формування магнітосом з магнітними наночастинками: MACG, MACGP, mms5, mms6, mms6-L, mamC, mamO, mms36, mms48, mamG, mamR, mamL, mamO, mamN, mamX, mamZ, mamY, mamJ, mamA, mamB, mamU, mamW, mad, man.

На рис. 1 по горизонталі представлено штами наступних магнітотаксисних бактерій: AMB-1, XM-1, MS-1, ME-1, SO-1, SP-1, BB-1, MSR-1, LBB-42, LBB-42, YQV-1, WYHC-3, MO-1, MC-1, IT-1, BW-2, MV-1, PR-1, SH-1, QH-2, LM-1, WYHS-4, DMHC-1, DMHC-6, THC-1, YQC-9, UR-1, YQC-3, YQC-5, XQGC-1, MYC-9, DMHC-8, SHHR-1, SS-5, RS-1, FSS-1, HK-1, BW-1, TM-1, MYR-1, XYR, YQR-1, LBB01, MYC-10, LBB02, XYC, WYHC-5, CS-04.

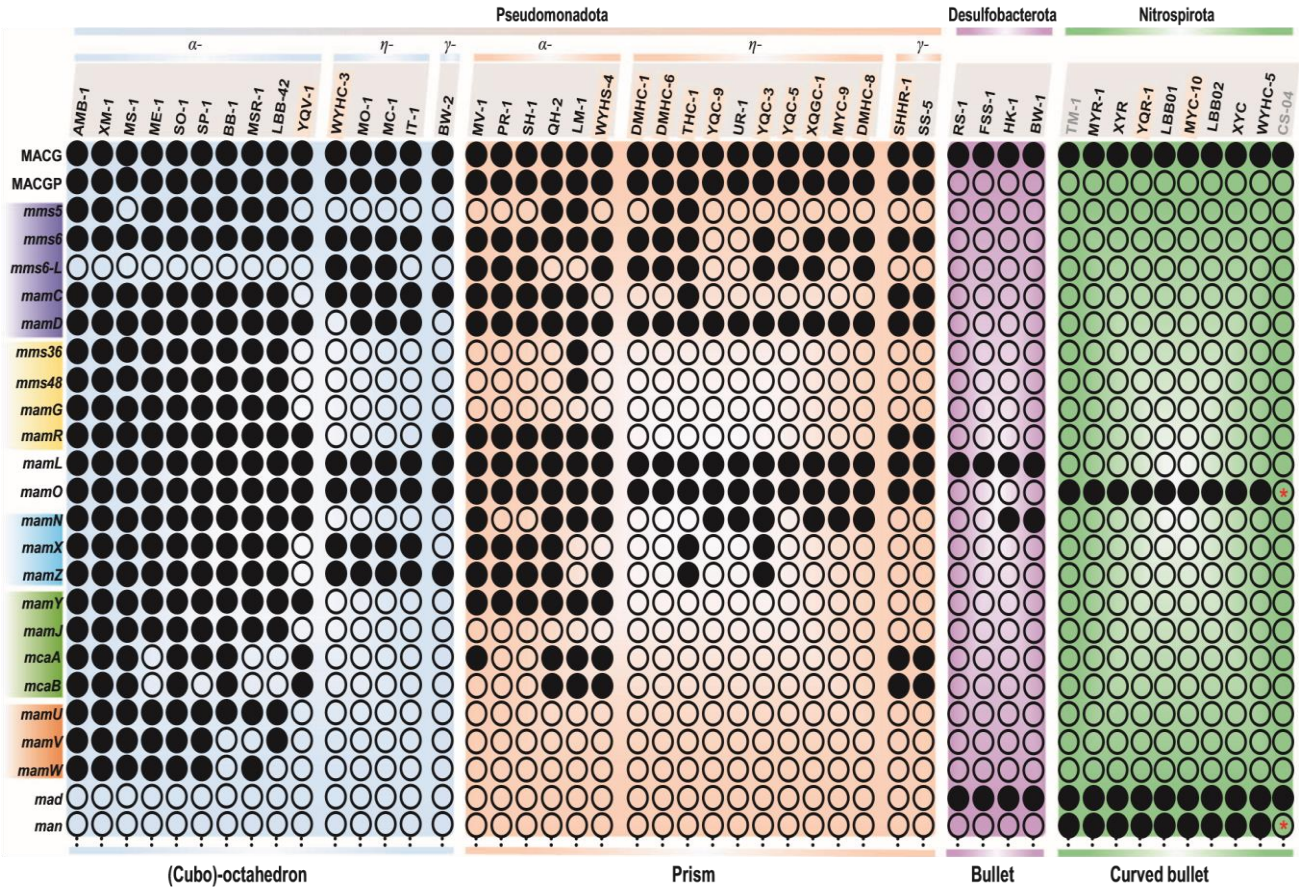

Рисунок 1 – Зв’язок між морфологією магнітних частинок і генами магнітосом. Гени (білки) з подібними функціями виділені однаковим кольором фону. Грецькі літери α, η та γ позначають класи Alphaproteobacteria, Candidatus Etaproteobacteria та Gammaproteobacteria. Штами МТВ показані в групах відповідно до їх філогенії та морфології кристалів магнетиту. Назви штамів, виділені жирним шрифтом і жовтим фоном. Чорні суцільні кружечки вказують на наявність відповідного гена; чорні порожнисті кола відповідають його відсутності. Чорні порожнисті кола з червоною зірочкою всередині вказують на те, що ген не був виявлений, можливо, через неповне секвенування геному. Дані роботи Peiyu Liu [6].

Для навчання класифікатора за даними роботи Peiyu Liu [6] (рис.1) виберіть ознаки (samples), тобто білки або набори білків, які можна використати для класифікації МТБ по формі їх біогенних магнітних наночастинок, а саме: MACG, MACGP, mamL, mam0, mad та сформуйте

текстовий файл «DecisionTree.txt» зі стовпчиками цих ознак (samples) для кожного штаму МТБ. В комірки файлу «DecisionTree.txt» на перетині штаму кожної МТБ та вибраних ознак (MACG, MACGP, mamL, mam0, mad) занесіть 1 (одиницю), якщо в таблиці (рис. 1) чорні суцільні кружечки, що вказують на наявність відповідного гену (білку), та занесіть 0 (нуль), якщо в таблиці (рис. 1) чорні порожнисті кола або чорні порожнисті кола з червоною зірочкою всередині, що відповідає відсутності гену (білку).

В файлі «DecisionTree.txt» сформуйте також стовпчик Shape, який буде представляти класи (тобто дані по формі БМН МТБ), за якими необхідно здійснити класифікацію (кластеризацію) МТБ. Всього на рис. 1 і відповідно в файлі «DecisionTree.txt» необхідно відобразити 47 ознак (білків або наборів білків), в тому числі: для класу Cubo-octahedron (кубоектаедр) – 15 ознак (білків), для Prism (призма) – 18 ознак, для Bullet (кулеподібна) – 4 ознаки та для Curved bullet (вигнута кулеподібна) – 10 ознак.

2.1.2.Перетворення нечислових даних в файлі з даними в числові

Для створення дерева рішень усі дані мають бути числовими, необхідно перетворити нечислові стовпці в числові значення. У пакеті pandas є метод map(), який працює зі словником з інформацією про те, як конвертувати значення.

Наприклад, {‘No': 0, ‘Weak': 1, ‘Severe': 2} – це означає перетворення значень "No" на 0, " Weak" на 1 та "Severe" на 2.

2.1.3.Створення об’єкту класу DecisionTreeClassifier модуля tree пакету sklearn

Застосуйте класифікатор «Дерево рішень» класу DecisionTreeClassifier модуля tree пакету sklearn для прийняття рішення на основі навчальних даних для задачі кластеризації МТБ за наступними чотирма класами, кожен з яких відповідає визначеній формі БМН: (Cubo)-octahedron – кубоектаедр, Prism –

призма, Bullet – кулеподібна та Curved bullet – вигнута кулеподібна для представлених на рис.1 магнітотаксисних бактерій.

2.1.4. Побудова дерева рішень та відображення дендрограми дерева рішень

Використайте метод fit() модуля tree пакету sklearn для побудови дерева рішень. Метод fit() модуля tree пакету sklearn повертає дані в спеціальному форматі, який необхідний для передачі цих даних методу plot_tree(), що відображає дендрограму дерева рішень. При побудові дендрограми з використанням іменованих аргументів дайте назви ознакам та класам, які відображають ознак та класів у файлі вихідних даних, та позначте різними кольорами окремі класи дерева рішень.

2.2. Результат роботи коду на мові програмування python для побудови дерева рішень

Результатом роботи коду на мові програмування python для побудови дерева рішень є дендрограма, в якій проведено розщеплення множини вихідних даних на підмножини (тобто класи) таким чином, щоб об’єкти кожної підмножини, що отримані в результаті цієї розбивки, були або представниками виключно одного класу, або кількість представників інших класів («домішок») була мінімальною. Це означає, що кількість об’єктів з інших класів, так званих «домішок», у кожному класі повинна прямувати до мінімуму в кожному вузлі дерева рішень.

Класи, які визначено однозначно в результаті побудови дерева рішень необхідно позначити різними кольорами.

На вузлах та листях дерева рішень необхідно назви ознак та класів позначити у відповідності до заданих у файлі з вихідними даними, а не у символьному вигляді.

3. ВИМОГИ ДО ОФОРМЛЕННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Розрахунково-графічна робота включає наступні файли:

1. _Title_page.txt – файл з титульною сторінкою оформлений, як показано в Додатку 1.
2. _DecisionTree.xlsx або _DecisionTree.txt – файл з вихідними даними, сформований, як описано в 2.1.1.
3. _MTB_shape_dendrogram.py – файл з кодом на мові програмування, який включає кроки описані в 2.1.2-2.1.4. В коді створити рядки документації, в яких описати функції коду, для кожного рядку коду дати рядок-коментар.
4. _Dendrogram.txt – файл з результатом роботи коду _MTB_shape_dendrogram.py, тобто зображенням дендрограми.

4. КРИТЕРІЇ ОЦІНЮВАННЯ РОЗРАХУНКОВО-ГРАФІЧНОЇ РОБОТИ

Згідно силабусу дисципліни «Прикладна біоінформатика», РГР, яка представлена у вигляді тесту оцінюється в 10 балів. Рейтингові (вагові) бали занять та рейтингові оцінки по всіх видах контролю в Google Класі дорівнюють відповідним балам в Силабусі з коефіцієнтом 10 для зручності розрахунку балів (щоб не використовувати дробові числа). Відповідно в кінці семестру всі набрані Вами бали будуть ділитися на 10, це стосується і РГР.

Таким чином РГР включає написання коду на мові програмування python для побудови дерева рішень та відповідь на наступні запитання:

1. На скільки класів відбулося розщеплення дерева рішень, побудованого на основі даних роботи [6]?

2. Які класи визначено однозначно в результаті побудови дерева рішень на основі даних роботи [6]?

3. Для яких класів значення параметру gini побудованого Вами дерева рішень буде дорівнювати 0?

4. Для яких класів значення параметру gini дерева рішень побудованого Вами дерева рішень буде більше 0 і менше 0.5?

5. Для яких класів значення параметру gini дерева рішень може дорівнювати 0.5?

6. Які параметри класу DecisionTreeClassifier використовуються для вимірювання якості розщеплення вузлів дерева рішень?

7. Які іменовані аргументи в методі plot_tree() Ви використали для кольорового позначення листя дерева рішень?

8. Який іменований аргумент в методі plot_tree() Ви використали для позначення назв ознак на вузлах та листях дерева рішень?

9. Який іменований аргумент в методі plot_tree() Ви використали для позначення назв класів на вузлах та листях дерева рішень?

10. Який метод пакету pandas використовується для перетворення нечислових даних у числові при роботі з таблицями пакету pandas DataFrame?

5. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Chetan Pandit, Arpita Roy, Suresh Ghotekar, Ameer Khusro, Mohammad Nazmul Islam, Talha Bin Emran, Siok Ee Lam, Mayeen Uddin Khandaker, David Andrew Bradley. Biological agents for synthesis of nanoparticles and their applications // Journal King Saud University – Science 34 2022.

2. Zborowski, M., and Chalmers, J. J. (2011). Rare cell separation and analysis by magnetic sorting. Anal. Chem. 83, 8050–8056. doi: 10.1021/ac200550d

3. Kraupner, A., Eberbeck, D., Heinke, D., Uebe, R., Schüler, D., and Briel, A. (2017). Bacterial magnetosomes – nature's powerful contribution to MPI tracer research. Nanoscale 9, 5788–5793. doi: 10.1039/C7NR01530E

4. Price, P. M., Mahmoud, W. E., Al-Ghamdi, A. A., and Bronstein, L. M. (2018). Magnetic drug delivery: where the field is going. Front. Chem. 6:619. doi: 10.3389/fchem.2018.00619

5. Van de Walle, A., Perez, J. E., Abou-Hassan, A., Hémadi, M., Luciani, N., and Wilhelm, C. (2020). Magnetic nanoparticles in regenerative medicine: what of their fate and impact in stem cells? Mater. Today Nano 11:100084. doi: 10.1016/j.mtnano.2020.100084

6. Liu, D., Hong, Y., Li, Y., Hu, C., Yip, T.-C., Yu, W.-K., et al. (2020). Targeted destruction of cancer stem cells using multifunctional magnetic nanoparticles that enable combined hyperthermia and chemotherapy. Theranostics 10, 1181–1196. doi: 10.7150/thno.38989

7. O. Gorobets, S. Gorobets, I. Sharai, T. Polyakova, V. Zablotskii, Interaction of magnetic fields with biogenic magnetic nanoparticles on cell membranes: Physiological consequences for organisms in health and disease, Bioelectrochemistry. 151 (2023)

8. S. Rosenfeldt, F. Mickoleit, C. Jörke, J. H. Clement, S. Markert, V. Jérôme, S. Schwarzinger, R. Freitag, D. Schüler, R. Uebe, and A. S. Schenk, Towards Standardized Purification of Bacterial Magnetic Nanoparticles for Future in Vivo Applications, Acta Biomater. 120, 293 (2021).

9. P. Goswami, K. He, J. Li, Y. Pan, A. P. Roberts, and W. Lin, Magnetotactic Bacteria and Magnetofossils: Ecology, Evolution and Environmental Implications, Npj Biofilms Microbiomes 8, 43 (2022).
10. E. M. Jefremovas, L. Gandarias, L. Marcano, A. Gacía-Prieto, I. Orue, A. Muela, M. L. Fdez-Gubieda, L. F. Barquín, and J. Alonso, Modifying the Magnetic Response of Magnetotactic Bacteria: Incorporation of Gd and Tb Ions into the Magnetosome Structure, Nanoscale Adv. 4, 2649 (2022).
11. A. Pekarsky, O. Spadiut. Intrinsically Magnetic Cells: A Review on Their Natural Occurrence and Synthetic Generation // Front. Bioeng. Biotechnol., 2020 Sec. Bioprocess Engineering, V. 8, 2020 |https://doi.org/10.3389/fbioe.2020.573183

6. РЕКОМЕНДОВАНІ ДЖЕРЕЛА

1. https://www.python.org/
2. https://scikit-learn.org/stable/user_guide.html
3. https://docs.scipy.org/doc/scipy/
4. https://pypi.org/project/pandas/
5. https://venngage.com/blog/what-is-a-decision-tree/
6. Peiyu Liu, Yue Zheng, Rongrong Zhang, Jinling Bai, Kelei Zhu, Karim Benzerara, Nicolas Menguy, Xiang Zhao, Andrew P. Roberts, Yongxin Pan and Jinhua Li/ Key gene networks that control magnetosome biomineralization in magnetotactic bacteria// National Science Review, Volume 10, Issue 1, 2023, nwac238 https://doi.org/10.1093/nsr/nwac238

Додаток А

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ» ІМЕНІ ІГОРЯ СІКОРСЬКОГО

Факультет біотехнології і біотехніки

Кафедра біоенергетики, біоінформатики та екобіотехнології

РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА

з дисципліни «Прикладна біоінформатика» на тему:

«Побудова класифікатора дерева рішень»

Науковий керівник:

д.т.н., проф. Горобець С.В.

Роботу виконав(-ла)

Студент(-ка), групи_____

_____

(прізвище та ініціали)

Київ 2024