**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА**

Факультет прикладної математики та інформатики

(повне найменування назва факультету)

Кафедра дискретного аналізу та інтелектуальних систем

(повна назва кафедри)

**Кваліфікаційна (бакалаврська) робота**

Порівняння оптимальності алгоритмів та методів класифікації машинного навчання

Виконала: студент групи ПМі-43с

спеціальності

122 «Комп’ютерні науки»\_\_\_

(шифр і назва спеціальності)

Бень Х. Б.

(підпис) (прізвище та ініціали)

Керівник Квасниця Г.А.

(підпис) (прізвище та ініціали)

Рецензент

(підпис) (прізвище та ініціали)

2024

Зміст

[Вступ 3](#_Toc168834349)

[Розділ 1. Теоретичні основи машинного навчання 5](#_Toc168834350)

[1.1 Означення основних понять 5](#_Toc168834351)

[1.2 Види класифікації 9](#_Toc168834352)

[1.3 Процес класифікації 12](#_Toc168834353)

[1.4 Оцінювання класифікації 14](#_Toc168834354)

[Розділ 2. Метод дерева рішень 18](#_Toc168834355)

[2.1 Теоретичні відомості 18](#_Toc168834356)

[2.2 Алгоритм побудови ID3 21](#_Toc168834357)

[2.3 Алгоритм побудови C45 23](#_Toc168834358)

[2.4 Алгоритм побудови CART 24](#_Toc168834359)

[2.5 Алгоритм побудови Chaid 25](#_Toc168834360)

[2.6 Алгоритм побудови Random Forest 26](#_Toc168834361)

[Розділ 3. Опис результатів 27](#_Toc168834362)

[3.1 Вибірки даних 27](#_Toc168834363)

[3.2 Аналіз метрик 29](#_Toc168834364)

[3.3 Візуалізація за допомогою Confusion Matrix 32](#_Toc168834365)

[3.4 Візуалізація за допомогою POC - кривої 33](#_Toc168834366)

[Висновок 34](#_Toc168834367)

[Джерела 36](#_Toc168834368)

Вступ

**Актуальність теми.** У наш час наплив інформації, завдяки можливостям її зберігання та отримання став, водночас, винятковою цінністю і великою складністю. Велика кількість інформації, можливість її доступного використання відкрили багато нових можливостей і дали поштовх для розвитку в різних сферах. Проте, це створило таку проблему, як невміння користуватись даними правильно. Такі обсяги інформації часто видаються складними для інтерпретації. Щоб їх коректно застосовувати потрібно вміти систематизувати для подальшого аналізу. Тому, необхідно вивчати різні способи їхнього впорядкування.

Актуальність теми з практичної точки зору полягає в тому, щоби досліджувати можливості класифікації даних і застосовувати для різних цілей та прикладів.

Питання класифікації у своїх роботах досліджувало багато вчених. Серед них:

* Френсіс Голтон (Francis Galton) – робота "Класифікація та вбудована асоціація" (Classification and Heredity);
* Рональд Фішер (Ronald Fisher) – праця "Статистичні методи для дослідження різниць та взаємозв'язків" (The Statistical Methods for Research Workers);
* Леонард Айзенберг (Leonard Eisenberg) – "Моделі класифікації та їх використання в розпізнаванні образів" (Classification Models and Their Use in Image Recognition);
* Том Мішель (Tom Mitchell): "Машинне навчання" (Machine Learning).

Проте, не було знайдено комплексної праці із порівняльним аналізом алгоритмів класифікації, тому бакалаврська робота є актуальною з теоретичної точки зору.

**Мета та завдання досліджень.** Метою бакалаврської роботи є порівняльний аналіз ефективності роботи алгоSSритмів класифікації на прикладі п’яти алгоритмів побудови дерева рішень: ID3, C45, CART, Chaid і Random Forest.

Досягнення мети передбачає:

* виокремлення критеріїв оцінки роботи алгоритмів;
* проведення дослідження роботи алгоритмів на основі обраних критеріїв;
* порівняння отриманих числових характеристик.

**Об’єкт дослідження** – алгоритми побудови дерева рішень: ID3, C45, CART, Chaid і Random Forest.

**Предмет дослідження** – оцінка ефективності роботи алгоритмів класифікації на прикладі п’яти алгоритмів побудови дерева рішень.

**Методи дослідження.** Алгоритми побудови дерева рішень: ID3, C45, CART, Chaid і Random Forest оцінено за допомогою критеріїв ефективності роботи та виокремлено область застосування кожного з цих алгоритмів.

Основні отримані результати є такими:

* виокремлені критерії оцінки роботи алгоритмів;
* на основі цих критеріїв було отримані метрики;
* з використанням метрик здійснено порівняльний аналіз оптимальності вибору алгоритму на певних наборах даних.

**Теоретичне значення** бакалаврської роботи полягає в узагальненні на базі алгоритмів дерева прийняття рішень критеріїв оптимальності оцінки роботи алгоритмів.

**Практичне значення.** Запропоновані у бакалаврській роботі критерії оцінки оптимальності роботи алгоритмів дерева прийняття рішень можуть бути використані у розділах курсу «Інтелектуальний аналіз даних» для студентів факультету прикладної математики та інформатики.

**Структура та обсяг бакалаврської роботи.** Бакалаврська робота складається зі вступу, трьох розділів, висновків і переліку літератури. Загальний обсяг бакалаврської роботи становить 33 сторінки, ?? рисунків, 6 таблиць і 15 найменувань бібліографічного списку.

# Постановка задачі

Машинне навчання та алгоритми класифікації знаходять широке застосування в різних галузях, від медицини до фінансів, допомагаючи аналізувати дані та приймати обґрунтовані рішення. Серед багатьох методів класифікації особливе місце займають алгоритми дерев рішень, які забезпечують створення моделей, що є легкими для розуміння та інтерпретації, що робить їх привабливими для використання у практичних задачах.

Основною метою цієї дипломної роботи є порівняння різних алгоритмів дерев рішень, таких як ID3, C4.5, CART, CHAID та Random Forest, на основі їх продуктивності за різними метриками. Для досягнення цієї мети необхідно виконати низку завдань, починаючи з дослідження теоретичних основ машинного навчання та класифікації. Важливо визначити основні поняття та методи класифікації, а також розглянути процес класифікації та способи оцінювання її результатів.

Наступним кроком є вивчення та опис алгоритмів дерев рішень. Це включає вивчення теоретичних основ кожного з розглянутих алгоритмів, а також детальний опис алгоритмів побудови дерев рішень ID3, C4.5, CART, CHAID та Random Forest. Цей етап допоможе зрозуміти внутрішні механізми роботи кожного алгоритму та їх особливості.

Практична частина роботи включає збір та підготовку вибірок даних для навчання та тестування моделей, реалізацію алгоритмів дерев рішень та оцінку їх продуктивності за такими метриками, як точність, чутливість, специфічність та F1-міра. Візуалізація результатів за допомогою матриці помилок та ROC-кривих дозволить більш наочно оцінити ефективність кожного з алгоритмів.

На основі отриманих результатів буде проведено порівняння алгоритмів за різними метриками, що дозволить визначити їхні переваги та недоліки у контексті розв'язуваних задач. Це дозволить зробити висновки щодо найбільш доречних методів для конкретних типів даних та задач класифікації.

Виконання цих завдань забезпечить ґрунтовне розуміння ефективності різних алгоритмів дерев рішень та допоможе обрати найбільш оптимальний метод для конкретних застосувань.

# Розділ 1. Теоретичні основи машинного навчання

## 1.1 Означення основних понять

Для глибинного розуміння класифікаційних процесів необхідний детальний розбір принципів і теоретичних понять.

**Машинне навчання**

Машинне навчання (Machine Learning, ML) – це галузь дослідження штучного інтелекту, пов’язана з розробкою та вивченням статистичних алгоритмів, які можуть навчатися на основі даних і узагальнювати невидимі дані, а отже виконувати завдання без чітких інструкцій.

Є три типи машинного навчання:

1. Навчання з учителем (Supervised Learning) – це процес створення моделі, що навчається на великій кількості прикладів для подальшого прогнозування результатів;
2. Навчання без вчителя (Unsupervised Learning) – це тренування алгоритму пошуку закономірностей, що пізніше дає змогу поділяти категорії;
3. Навчання з підкріпленням (Reinforcement Learning) – це процес навчання моделі через взаємодію з середовищем, і як наслідок отримує покарання за невдачі або нагороди за здобутки.

Найчастіше застосовують у таких галузях:

1. Опрацювання мови;
2. Розпізнавання образів;
3. Аналіз даних;
4. Рекомендаційні системи;
5. Виявлення аномалій.

Найвідомішими алгоритмами МL вважаються: класифікація, кластеризація, регресія та нейронні мережі.

**Класифікація**

Класифікація – це процес розподілення елементів предметної області на конкретні підгрупи за певним алгоритмом, що формується визначальними характеристиками.

Завдання класифікації – це віднесення об’єкту до того чи іншого класу. Такий тип поділу об’єктів, що має наперед визначені класи та повинен розрізнити елементи між ними, відносять до «класифікації з учителем».

Основна мета класифікації полягає в тому, щоб створити модель, яка може точно призначити мітку або категорію новому спостереженню на основі його характеристик.

Методи поділяють на такі групи:

1. Лінійні методи - дискримінантний аналіз (LDA): Використовується для розділення двох або більше класів, оптимізуючи роздільну гіперплощину.
2. Підтримуючі вектори (SVM): Знаходять гіперплощину, що максимально розділяє класи,
3. К-найближчих сусідів (k-NN): Класифікує об'єкт на основі голосування його найближчих сусідів у просторі ознак.
4. Методи на основі дерев рішень: Створюють дерево-модель на основі тренувальної вибірки
5. Байєсівські методи: Використовують теорему Байєса для класифікації, з припущенням незалежності ознак
6. Методи ансамблевого навчання: Комбінують декілька моделей для підвищення точності та стійкості
7. Методи на основі правила асоціацій: Виявляють зв'язки між змінними у великих базах даних для класифікації.

**Кластеризація**

Проте не завжди ми отримуємо сформовані класи, за якими сортувати. В такому випадку ми отримуємо задачу «класифікації без учителя». Відповідно нам потрібно сформувати класи, розділивши заданий набір об’єктів на групи за певними критеріями, що означає «здійснити кластерний аналіз».

Кластеризація – це метод аналізу даних, який використовується для групування схожих об'єктів або прикладів у наборі даних.

Основна ідея полягає в тому, щоб об'єкти, які схожі один на одного, були розташовані в одному кластері, тоді як об'єкти, які відрізняються, були розташовані у різних кластерах.

Мета кластерного аналізу : виявити закономірності та структури в наборі даних, які можуть надати розуміння базових зв’язків і асоціацій, що стане основою поділу на кластери.

Алгоритми кластерного аналізу розрізняють на такі категорії:

* Ієрархічні алгоритми (Hierarchy Algorithms);
* K-means;
* Алгоритми на основі щільності (Density-based methods);
* Модельно-орієнтовані алгоритми;
* Нечітка кластеризація (Fuzzy Clustering).

Кластерний аналіз застосовують у різних сферах і галузях. Ось кілька типових прикладів:

1. економіка, сегментація ринку – розділення клієнтів на групи на основі їхньої купівельної, демографічних чи інших характеристик;
2. біологія та медицина – групування пацієнтів із подібними клінічними характеристиками;
3. менеджмент – групування персоналу для подальшого вибору стратегії управління;
4. соціологія – поділ респондентів на групи.

**Регресія**

Регресія – це статистичний метод, який використовується для моделювання взаємозв’язків між залежною змінною (цільовою змінною) та однією чи декількома незалежними змінними (прогнозованими змінні).

Метою використання варто виокремити прогнозування – передбачення значення безперервної змінної на основі отриманого набору даних.

Можна виділити такий поділ типів регресії:

* лінійна регресія;
* логістична регресія;
* поліноміальна регресія;
* багатофакторна регресія;
* регресія з регуляризацією.

Оцінювати коректність моделі можна за такими метриками:

1. середньоквадратична помилка (MSE): середня абсолютна різниця між прогнозованим і фактичним значеннями цільової змінної;
2. середня абсолютна помилка (MAE): середня квадратична різниця між прогнозованим і фактичним значеннями цільової змінної;
3. коефіцієнт детермінації (R²): Показує, яка частка варіації цільової змінної пояснюється моделлю.

Застосовують регресію часто у таких випадках:

* оцінювання цін на продукт;
* прогнозування продажів;
* моделювання залежностей;
* виявлення факторів ризику.

## 1.2 Види класифікації

Можна розрізняти такі види класифікації:

1. *За ознаками:*
2. **допоміжна (штучна) класифікація** – відбувається з використанням якоїсь зовнішньої ознаки, яка не є суттєвою ознакою, але оптимальна для застосування, щоб надати порядок предметній області. Наприклад: відсортовування учнів за алфавітом – ми можемо швидко відшукати необхідне прізвище, що для деяких завдань є зручно, але це не дає нам інформативного наповнення про вміння та якості особи;
3. **природна класифікація** – це складніший варіант способу впорядкування, оскільки потребує вивчення та аналіз властивостей об’єкту. Завданням є пошук закономірностей між елементами для виокремлення їхніх властивостей. Ґрунтується на природніх зв’язках і подібностях чи характерних особливостях. Також може використовуватися для передбачення, якої категорії будуть стосуватися нові дані. Наприклад, класифікація для формування видів тварин чи рослин у біології [1].
4. *За процедурою розподілу:*
5. **проста** – використовує невелику кількість категорій або класів для розподілу об'єктів. Наприклад, може бути використана бінарна класифікація, де об'єкти поділяються на два взаємовиключні класи. Це простий підхід, який дає можливість зрозуміти основний аспект або властивість об'єктів, але може бути недостатнім для більш деталізованого аналізу або розпізнавання складних взаємозв'язків. Може бути ефективною, якщо основна інформація може бути висвітлена в межах обмеженого набору класів;
6. **складна** – використовує більшу кількість категорій або класів для розподілу об'єктів. Цей підхід дозволяє враховувати більшу різноманітність характеристик і відтворювати більш деталізовану структуру або відносини між об'єктами. Складна класифікація може включати багатокласову класифікацію з численними категоріями або природну ієрархічну класифікацію, де класи організовані у вигляді дерева або ієрархії з різними рівнями. Може бути необхідною, коли варто враховувати багато різноманітних характеристик або з'ясувати багатоаспектну залежність між об'єктами.
7. *За кількістю ознак:*
8. **одновимірна** – використовує лише одну ознаку або змінну для розподілу об'єктів на класи. Цей підхід простий та прямолінійний, оскільки базується на аналізі лише одного аспекту об'єктів. Наприклад, можна класифікувати студентів на основі їхнього віку, де єдиним критерієм є числове значення віку. Одновимірна класифікація може бути корисною, коли важлива лише одна особливість або коли інші ознаки не мають значення для вирішення конкретного завдання. Тобто є ефективнішою для наборів даних із простим сценарієм;
9. **багатовимірна** – використовує кілька ознак або змінних для розподілу об'єктів на класи. Цей підхід дозволяє враховувати більш широкий набір характеристик об'єктів, що надає більше гнучкості у випадках, коли важливо врахувати багато аспектів. Наприклад, можна класифікувати студентів за віком, середнім балом, кількістю годин, витрачених на вивчення і т. д. Кожна з цих ознак може бути використаною як вимір для розподілу об'єктів у багатовимірному просторі. Багатовимірна класифікація дозволяє отримати більш повну інформацію про зв'язки та структуру даних, а також може покращити точність класифікаційної моделі для оптимальнішого аналізу даних.
10. *За метою:*
11. **прогнозування** – застосовується для передбачення результату від надходження нових даних. Наприклад, прогнозування приросту попиту для певних товарів;
12. **ідентифікація** – вирізнення, якої з категорій стосується конкретний елемент за його характеристиками. Прикладом може бути ідентифікація осіб за їхніми біометричними даними;
13. **систематизація** –використовується для організації та впорядкування даних, дає можливість групувати схожі об'єкти разом, що полегшує подальше оброблення. Наприклад, формування баз даних і створення спрощеної системи пошуку елементів;
14. **виявлення аномалій** –відокремлення елементів від нормального розподілу.Може застосовуватися, щобивідокремити нормальну активність від підозрілої активності в мережі або для виявлення шахрайських операцій у банківських рахунках;
15. **пріоритезація** –необхідна для ранжування об’єктів за їхньою релевантністю. Може використовуватись для формування порядку рекомендацій веб-сторінок після запиту у пошукових системах.

## 1.3 Процес класифікації

Види наборів вхідних даних поділяють на дві підмножини: навчальну і тестову.

**Навчальна множина (або training set)** –це набір даних, який використовується для навчання моделі. Вона містить вхідні дані (приклади) та відповідні вихідні значення (цільові значення). Вихідні значення використовуються для налаштування параметрів моделі так, щоб вона могла розпізнавати патерни та робити передбачення.

**Тестова множина (або test set)** також включає вхідні дані (приклади) та відповідні вихідні значення. Вихідні значення використовуються для оцінки працездатності моделі на нових, невидимих даних.

Варто описати в якому порядку відбувається класифікація. Процес складається з двох етапів: конструювання моделі та її використання.

1. **Конструювання моделі.** На цьому етапі відбуваються підготовчі процеси для формування математичної формули чи тренувальної моделі, до яких пізніше застосовується конкретний алгоритм. Тобто ми описуємо множину визначених класів з такими кроками:

1. ***збір та підготовка даних*** – необхідно зібрати відповідні дані, які будуть використовуватися для класифікації. Це можуть бути числові дані, текстові описи, зображення, аудіо або будь-який інший тип даних. Потім дані підлягають попередньому обробленню, включаючи очищення, перевірку на коректність та масштабування;
2. ***визначення та пріоритезація ознак*** – необхідно вибрати набір ознак, що описують об'єкти, та визначити, які мають найбільше значення. Вибір може відбуватися або вручну, або враховуючи результати обчислення цінності інформації, що несе кожна ознака;
3. ***вибір алгоритму*** – вибір алгоритму залежить від типу даних та завдання класифікації. Якщо нам необхідно робити вибір, враховуючи велику кількість аспектів, то краще вибирати алгоритми машинного навчання, а якщо ми будемо працювати з математичними обчисленнями – статистичного;
4. ***тренування моделі*** – після визначення алгоритму необхідно навчити модель на тренувальному наборі даних. Тренування моделі включає пошук оптимальних параметрів алгоритму, які найкраще адаптуються до навчальних даних. Це може включати процес навчання з учителем, де дані мають мітки класів, або навчання без учителя, де дані не мають міток, і модель виявляє внутрішні закономірності або групи в даних;
5. ***валідація моделі*** – після тренування моделі необхідно перевірити її ефективність та точність на валідаційному наборі даних. Це допомагає оцінити якість класифікації та зробити необхідні налаштування для поліпшення моделі.

2. **Використання моделі**. Етап, на якому відбувається власне поділ об’єктів предметної області. Робота відбувається з новими або тестовими даними, до яких ми застосовуємо нашу натреновану модель. Рівень точності – відсоток правильно класифікованих прикладів у тестовій множині. Можна виділити такі етапи:

1. ***тестування моделі*** – після успішної валідації модель може бути протестована на тестовому наборі даних, які модель раніше не бачила. Це дозволяє оцінити загальну здатність моделі до класифікації нових об'єктів та отримати бажаний результат;
2. ***оцінка правильності*** – нам необхідно отримати підтвердження коректності результатів. Для цього розраховується рівень точності;
3. ***застосування моделі*** – після успішного тестування модель може бути застосована для класифікації нових невідомих об'єктів або даних в реальному часі.

Процес класифікації може бути ітеративним, і вимагати налаштування параметрів моделі та оптимізацію для досягнення кращої точності класифікації.

## ***1.4 Оцінювання класифікації***

Є чотири типи оцінювання якості моделі класифікації:

1. **крос-перевірка (Cross-Validation)** –використовує один початковий набір даних, який поділяється на кілька підмножин. Модель навчається на одних підмножинах, а оцінка її ефективності проводиться на інших. Цей процес повторюється кілька разів з різними комбінаціями. В результаті отримується усереднена метрика ефективності моделі. Цей спосіб варто використовувати для вибірок з великою кількістю даних, щоби отримати різні комбінації навчальної та тестової вибірок [5];
2. **тестова множина (Test Set)** – це оцінювання, яка на початку отримує два чітко поділені набори даних: Тестова вибірка – це окремий набір даних, який не використовується під час навчання моделі для обчислення точності. Модель, яка була навчена на тренувальному наборі, використовується для класифікації об'єктів у тестовій множині;
3. **таблиця спряженості (Confusion Matrix)** – це інструмент для оцінки точності моделі класифікації. Її елементами у головній діагоналі є кількість правильно класифікованих об'єктів для кожного класу та неправильно класифікованих – на побічній. Має такі складові:

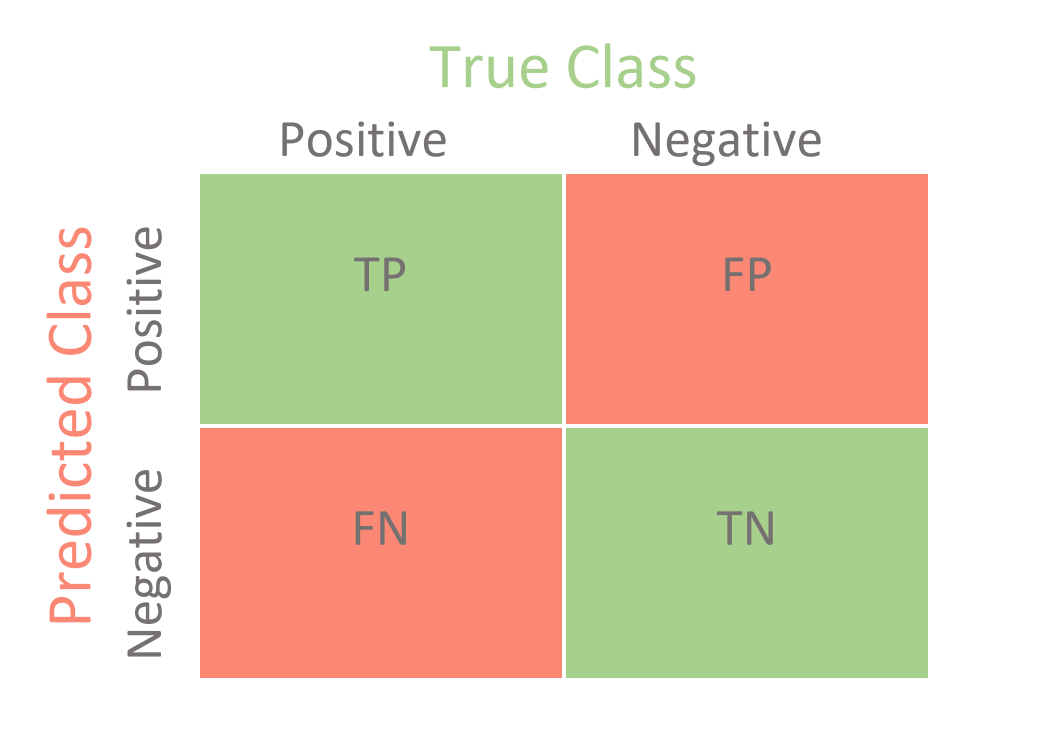


Рис. 1. Таблиця спряженості

* TP (True Positive) – кількість передбачень, де класифікатор правильно прогнозує позитивний клас (істинно-позитивні випадки);
* TN (True Negative) – кількість передбачень, що мають правильно визначений негативний клас (істинно-негативні випадки);
* FN (False Negative) – кількість передбачень, де помилково визначений негативний клас як позитивний. Це помилка 1-го роду (хибно-негативні зразки);
* FP (False Positive) – кількість передбачень, де помилково визначений позитивний клас як негативний. Це помилка 2-го роду (хибно-позитивні випадки) [англ. матриця].

Як наслідок, з цієї таблиці можна вивести такі показники, як:

* **точність (accuracy)** – дає загальну точність моделі, тобто частку правильно класифікованих від загальної кількості зразків. Для розрахунку використовують таку формулу:

AC = (TP+TN)/(TP+TN+FP+FN);

* **коефіцієнт неправильної класифікації (Misclassification Rate)** – повідомляє, яка частка прогнозів була неправильною. Також відомий, як помилка класифікації. Можна обчислити його за допомогою виразу:

MR = (FP+FN)/(TP+TN+FP+FN) або (1-AC);

* **правильність (precision)** – несе в собі інформацію, яка частка прогнозів позитивного класу була насправді позитивною. Щоб обчислити точність, використовують наступну формулу:

PR = TP/(TP+FP);

* **чутливість (recall/sensitivity)** – дає інформацію про те, яку частку всіх позитивних зразків класифікатор правильно передбачив як позитивні. Він також відомий як **істинний позитивний коефіцієнт (TPR)**. Для обчислення використовують таку формулу:

SE=TPR= TP/(TP+FN);

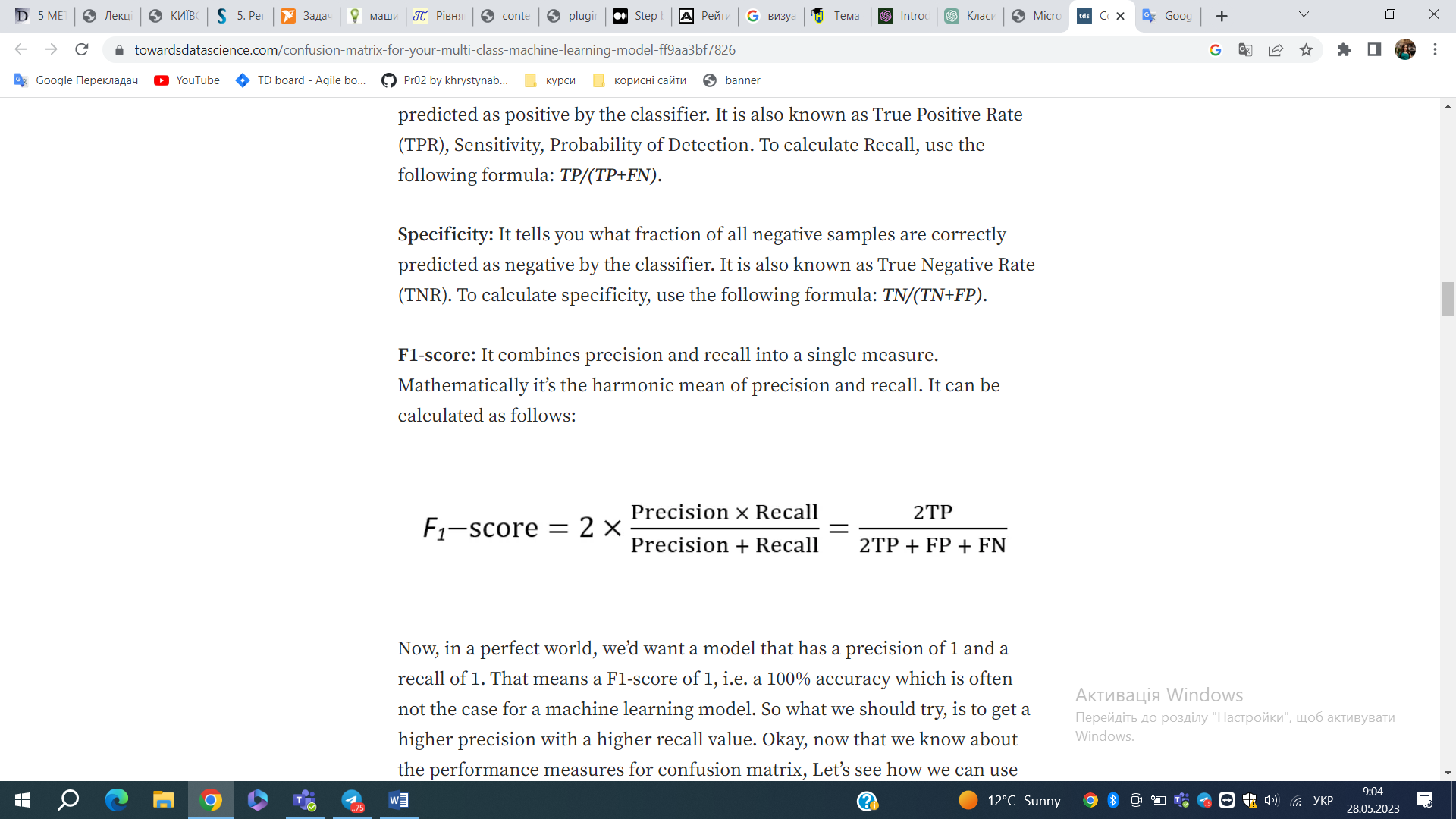
* **хибний позитивний коефіцієнт (FPR)** –частка, яка обчислюється такою формулою:

FPR = FP/(TN+FP);

* **специфічність (specificity)** – це частка всіх негативних зразків, які класифікатор правильно передбачив як негативні. Він також має назву – справжній **негативний коефіцієнт (TNR).** Для його розрахунку маємо формулу:

SP =TNR=TN/(TN+FP);

* **F1-міра (F1-score)** – поєднує в собі точність і запам’ятовування. Математично – це середнє гармонійне значення точності та запам’ятовування. Його можна розрахувати наступним чином:

;

1. **ROC-аналіз (Receiver Operating Characteristic)** – відбувається завдяки аналізу результатів внаслідок побудови ROC-кривої. Він показує залежність між чутливістю (доля правильно визначених позитивних об'єктів) та специфічністю (доля правильно визначених негативних об'єктів) за зміни порогу. Крива ROC може бути використана для порівняння різних моделей та вибору оптимального порогу відсічення для заданої задачі класифікації.

ROC-крива будується як графік, де на осі X відображається специфічність, а на осі Y – чутливість. Для кожного порогового значення обчислені метрики відображаються на графіку, і відрізок з'єднує ці точки.

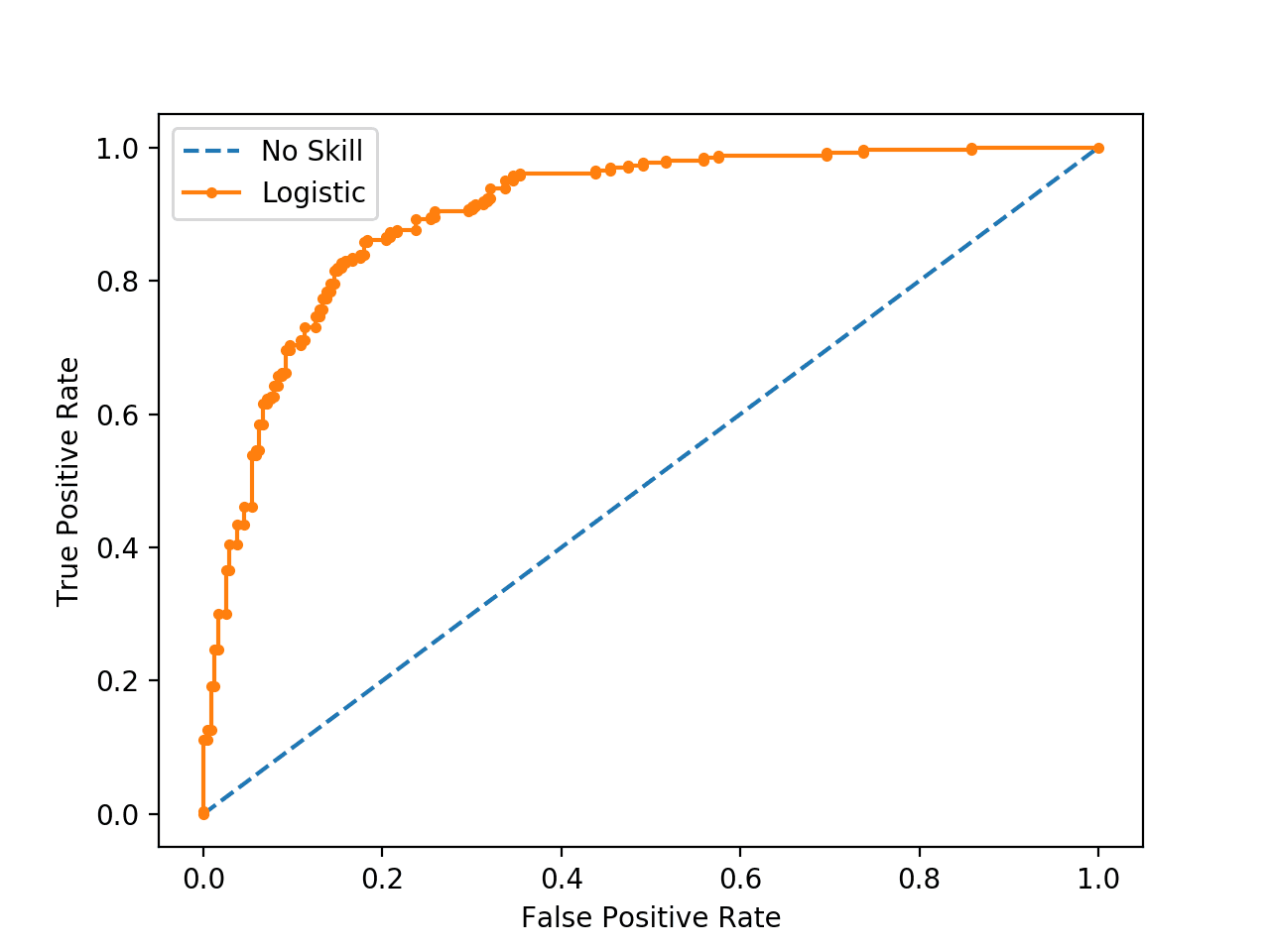
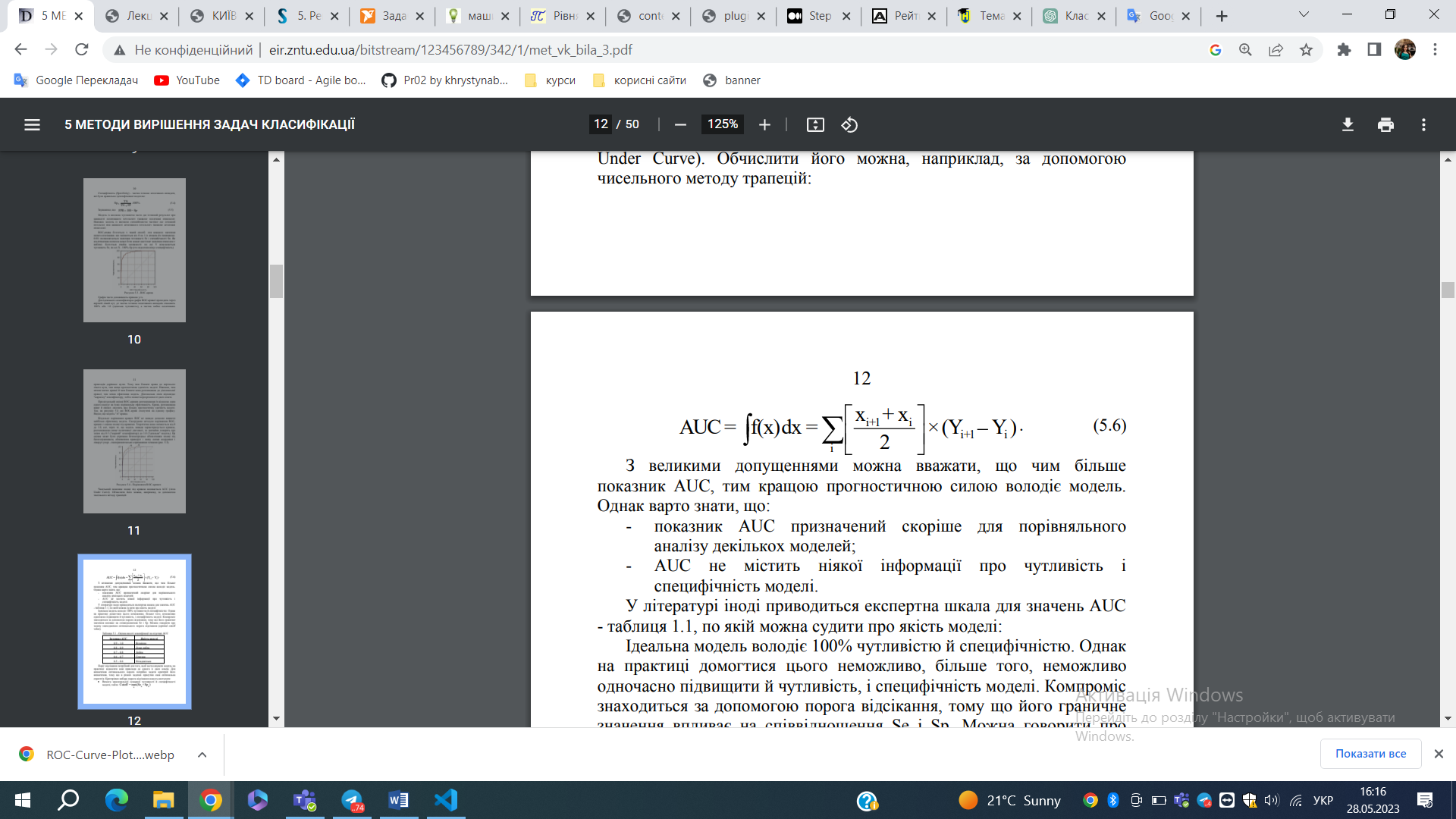


Рис. 2. ROC-крива

Точність може бути оцінена за допомогою площі під кривою (Area Under the Curve – AUC). Чим більше AUC, тим краще модель може розрізнити класи [7]. AUC обчислюється за допомогою виразу:

.

Пояснення

# Розділ 2. Метод дерева рішень

## 2.1 Теоретичні відомості

Дерево рішень вважається одним із найпопулярніших алгоритмів для розв’язання задач класифікації. Його також використовують і для прогнозування – передбачення майбутніх результатів. Відмінність полягає у значенні цільової змінної, якщо вона є дискретною, то метод застосовується для задачі класифікації, якщо ж змінна безперервна, тоді для прогнозування.

Дерево рішень – це структура даних, яка використовується для прийняття рішень, базуючись на результатах моделі, сформованої внаслідок попередньо опрацьованої сукупності даних.

Основні компоненти дерева рішень:

* вузол (Node) – демонструє умову та розділення внаслідок неї;
* кореневий вузол (Root Node): перший вузол дерева, з якого починається розділення даних;
* листок (Leaf) – знаходяться внизу дерева, представляє кінцеві рішення або прогнози;
* гілка – вказують на можливі значення ознак.

Дерево рішень зберігає ієрархічну структуру, яка створюється набором правил, що описують логічні конструкції «якщо: … то: …».

Одним із часткових випадків дерева рішень є бінарне дерево, що є прикладом простої структури, оскільки відповіді можуть бути лише «так», «ні».

Як результат побудови схеми у вигляді дерева можна отримати аналіз даних і спосіб пошуку відповідей на конкретні запитання. Формулювати схеми можна за допомогою дуже різних допоміжних даних, що мають прямий стосунок до поставленого запитання. Чим більше надати інформації, умов, атрибутів, тим точніше можна отримати результат, оскільки натренована модель буде мати в собі ширший обсяг опрацьованих обставин.

Процес побудови та формування структури, а також процес знаходження результатів буде описано у наступних підрозділах детально для кожного досліджуваного алгоритму.

У підсумку варто описати переваги та недоліки використання даного алгоритму.

*Переваги дерева рішень:*

1. легко зрозуміти і інтерпретувати – дерева рішень надають простий і зрозумілий спосіб прийняття рішень. Вони можуть бути інтерпретовані і пояснені людям без спеціальних знань в галузі машинного навчання;
2. використовується для класифікації та регресії – дерева рішень можуть використовуватись як для задач класифікації, так і для задач регресії. Вони можуть прогнозувати категоріальні мітки або числові значення;
3. може працювати зі змішаними типами даних – дерева рішень можуть опрацьовувати набори даних, які містять як категоріальні, так і числові ознаки. Вони можуть автоматично обробляти такі типи даних без потреби в додаткових попередніх перетвореннях.

*Недоліки дерева рішень:*

1. схильність до перенавчання – дерева рішень можуть стати складними та перенавчатися на тренувальних даних, особливо якщо глибина дерева занадто велика або використовуються інші параметри, які сприяють складності моделі. Це може призводити до поганої універсальності та низької здатності до узагальнення на нові дані;
2. вразливість до шуму та змін в даних – дерева рішень можуть бути чутливими до шуму та випадкових змін в даних. Навіть невеликі зміни в тренувальних даних можуть призводити до істотних змін у структурі та правилах прийняття рішень, що може негативно впливати на результат.

Для побудови дерева рішень використовується багато алгоритмів, таких як:

* ID3;
* CART;
* C4.5;
* C5.0;
* NewId;
* ITrule;
* CHAID;
* CN2;
* Cart;
* Random forest.

Спектр застосувань цього алгоритму також дуже широкий, оскільки необхідність роботи з даними зараз присутня у більшості сфер.

Відомими прикладами застосування можна назвати такі напрямки:

1. банківська справа – сформована сукупність даних клієнтської бази, що потребує автоматизованого аналізу про надання чи відхилення кредиту;
2. медицина – діагностика захворювань різної складності, допомога у пошуку діагнозів;
3. біологія – пошук взаємозв’язків між біологічними ознаками для формування структури класифікації видів.

Це лише частина таких можливостей, які вже використовують. Проте цей список буде суттєво розширюватися, зокрема, у цій роботі буде описано як ще можна корисно застосувати такі методи.

## 2.2 Алгоритм побудови ID3

ID3 (Iterative Dichotomizer 3) – це один із найбільш вживаних алгоритмів побудови дерева рішень. Назва походить від опису процесу, оскільки алгоритм ітеративно (повторно) дихотомізує (поділяє) ознаки на дві або більше групи на кожному етапі [2].

Алгоритм був винайдено Россом Квінланом, австралійським науковцем, наприкінці 1970-х і на початку 1980-х років. Квінлан відомий вагомим внеском у сфері машинного навчання, зокрема внаслідок серії методів класифікації.

Ідея алгоритму полягає в тому, щоб знайти атрибут, який на даному етапі є найбільш корисним для класифікації набору даних. Пошук найкориснішого атрибута відбувається завдяки обрахунку приросту інформації, що залежить від значення ентропії.

Ентропія (Entropy) – це міра невпорядкованості або невизначеності в наборі даних. Називають ще «ступенем хаосу».

Це означає, що ентропія використовується для пошуку найінформативнішого атрибута, що сприяє зменшенню невизначеності. Чим менша ентропія, тим менший показник невизначеності, тобто метою є мінімізувати ентропію. Якщо ентропія дорівнює нулю, то всі елементи належать до одного класу, і ми маємо «зрозумілий» результат, якщо – одиниці, то елементи все ще хаотично розділені між різними класами [1].

Розрахунок ентропії відбувається за такою формулою:

,

де:

* n = кількість класів;
* A – приклад значення категорії одного із атрибутів;
* = це показник ймовірності, що обраховується як частка від кількості елементів класу та загальної кількість рядків обраної категорії.

Приріст інформації (Information Gain) – це показник, що відповідає за вибір атрибута, за яким буде відбуватись поділ даних. Він шукає найвагоміший атрибут, що найкраще зменшує невизначеність та класифікує цільові змінні [2].

Обчислення інформації відбувається за такою формулою:

,

де:

* m – кількість категорій;
* S – атрибут, тобто риса, яку характеризують дані у стовпці.

Тоді, приріст інформації обчислюємо за допомогою виразу:

,

де H(D) – ентропія всього набору даних.

**Хід роботи алгоритму**:

1. Початок алгоритму:

* Алгоритм починається з виклику функції *id3(train\_data, label),* у яку передається вибірка із тренувальними даними та цільова змінна.
* Створюється порожнє дерево
* Формується список унікальних класів *class\_list* в навчальних даних.

1. Пошук найінформативнішої характеристики:

* Викликається функція *find\_most\_informative\_feature(train\_data, label, class\_list)*, яка шукає характеристику з найвищим приростом інформації.
* Ця функція використовує функцію *calc\_info\_gain* *(feature, train\_data, label, class\_list)* для обчислення приросту інформації для кожної характеристики.

1. Функція *calc\_info\_gain* є необхідна для для обчислення приросту інформації для кожної характеристики. Під час її роботи викликається для кожної ознаки функція *calc\_entropy(feature\_value\_data, label, class\_list)* для підрахування ентропії, а також функція для обчислення загальної ентропії *calc\_total\_entropy()*.
2. Побудова піддерева для обраної характеристики:

Обрана характеристика передається до функції generate\_sub\_tree(feature\_name, train\_data, label, class\_list), яка будує піддерево для цієї характеристики.

1. Рекурсивна побудова дерева:

* Функція *generate\_sub\_tree()* рекурсивно викликається для кожного унікального значення характеристики, поки не буде побудовано повне піддерево для обраної характеристики.
* Кожне піддерево обробляється за допомогою рекурсії у функції *make\_tree().*

1. Завершення роботи алгоритму:

Після завершення рекурсії, алгоритм повертає побудоване дерево tree.

Для передбачення результатів, завдяки створеному дереву, необхідно зробити такі кроки:

1. починаємо із коренового вузла;
2. перевіряємо значення атрибута;
3. рухаємося по відповідній гілці;
4. повторюємо кроки рекурсивно;
5. алгоритм закінчується, коли доходимо до конкретного результуючого класу.

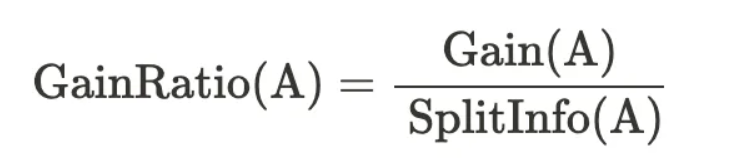
Отже, у даному розділі було описано особливості та роботу алгоритму ID3. Результати, переваги, недоліки будуть описані у наступному розділі.

## 2.3 Алгоритм побудови C45

C45 – це покращена версія алгоритму ID3, автором якої є також Росс Квінлан.

Оскільки алгоритм базується на ID3, то зберігаються критерії поділу та такі поняття, як ентропія та приріст інформації. Проте, є декілька важливих функцій, що суттєво покращують роботу цього алгоритму:

1. опрацювання неперервних атрибутів – це означає, що алгоритм C45 може працювати не тільки з дискретними значеннями, але й з неперервними. Це відбувається завдяки утворенню порогових значень після обчислення середніх значень між парами сусідів. Після чого для порогів обчислюється ентропія та приріст інформації;
2. опрацювання відсутніх значень – після знайдення відсутнього значення відбувається обчислення ймовірності для заміни найдоречнішим у даному контексті значенням;
3. коефіцієнт співвідношення приросту функції (Gain Ratio) – покращений показник для збалансування інформаційного приросту, щоб не надавати перевагу атрибутам із більшості кількістю рис, оскільки вони мають меншу ентропію, обчислюють за формулою:

****;

1. обрізання дерева – це покращення для уникнення перенавчання дерева, що відбувається внаслідок видалення розгалужень із несуттєвим впливом на результат, перевіряючи чи підходять умови відхилення.

Цікавий факт, що існують нові покращені версії цієї серії алгоритмів. Наприклад, відомо про C5.0, проте його деталі не є доступними публічно, оскільки Квінлан його використовує комерційно.

**Хід роботи алгоритму:**

1. Ініціалізація дерева рішень:

Створюється клас *Node*, який представляє вузли дерева рішень. Він має атрибути, такі як *attribute* (атрибут для розділення), *threshold* (поріг розділення), *label* (мітка класу для листків) та *is\_leaf* (прапорець, що позначає листки).

1. Розрахунок ентропії:

Функція *entropy(y)* обчислює ентропію для масиву класів y.

1. Розрахунок коефіцієнта приросту інформації:

Функція *information\_gain\_ratio(y, y\_subsets)* обчислює коефіцієнт приросту інформації між вершинами дерева для визначення найкращого розбиття даних на кожному кроці.

1. Розділення даних:

Функція *split\_data(X, y, attribute, threshold)* розділяє навчальні дані X та y за вказаним атрибутом та порогом розділення.

1. Вибір найкращого розбиття:

Функція *best\_split(X, y)* визначає найкраще розбиття для даного набору даних, шукаючи атрибут і поріг розділення з найбільшим коефіцієнтом приросту інформації.

1. Побудова дерева:

Функція *c45(X, y, min\_samples\_split=2, depth=0, max\_depth=None)* рекурсивно будує дерево рішень на основі навчальних даних X та y.

1. Обрізка дерева

prune\_tree(node, X, y) яка використовується для обрізки дерева після побудови. Функція порівнює помилку зі зрізом дерева та без нього, і якщо зріз не покращує точність, то він видаляється.

1. Передбачення результатів:

* Функція *predict\_single(node, x)* передбачає клас для конкретного запису x з допомогою побудованого дерева *node*.
* Функція *predict(tree, X)* використовує *predict\_single* для передбачення класів для всього набору даних X.

## 2.4 Алгоритм побудови CART

Алгоритм CART (Classification and Regression Trees) – це метод машинного навчання, який використовується як для задач класифікації, так і для регресії [4].

Алгоритм був заснований у 1984 році Лео Бріманом, Джером Фрідманом та Карл Дж. Стоуном.

Загальний принцип роботи алгоритму є таким, як і у попередніх алгоритмах, проте, суттєвою відмінністю є критерій поділу. Для задач класифікації таким критерієм є індекс Джині, а для задач регресії – MSE (Mean Squared Error).

Індекс Джині – це показник, що демонструє, наскільки добре атрибут розділяє тренувальні дані за класами. Тобто, це метрика для пошуку найкращої однорідності для підвузлів [5]. Цей показник зберігає значення суми квадратів ймовірностей для кожного класу. Нижче наведемо формулу для його обчислення:

*,*

де – це ймовірність віднесення об’єкту до певного класу, n – кількість підкласів і S – вузол, для якого відбувається обчислення.

**Хід роботи алгоритму**:

1. Початкові налаштування:

Алгоритм починається зі створення класу *TreeNode*, який представляє вузол дерева рішень.

1. Вибір кращого розбиття:

Для кожного вузла дерева виконується процедура вибору кращого розбиття за допомогою функції *get\_best\_split*.

* Визначення класів: Спочатку визначаються всі унікальні класи в наборі даних.
* Ітерація по всіх ознаках: Для кожної ознаки та її значень розраховується Gini Index за допомогою функції *calculate\_gini\_index (groups, classes).* Ця функція обчислює рівень неоднорідності даних після розбиття. Найкраще розбиття визначається мінімальним значенням Gini Index.

1. Розбиття даних:

Функція *split\_dataset(index, value, dataset)* розбиває дані на дві групи (ліву та праву) за обраним атрибутом та пороговим значенням.

1. Рекурсивне розбиття:

Після визначення найкращого розбиття, дані рекурсивно розбиваються на підмножини за допомогою функції *split(node, max\_depth, min\_size, depth).*

* Листові вузли: Якщо вузол є листовим (тобто не може бути розділеним далі), його значення встановлюється за допомогою функції *to\_terminal(group).*

1. Побудова дерева:

Функція cart починає побудову дерева рішень, викликаючи *get\_best\_split(dataset)* для кореневого вузла, а потім рекурсивно розбиває дані за допомогою split.

1. Прогнозування:

Прогнозування для нового прикладу виконується за допомогою функції *predict.*

Це здійснюється шляхом рекурсивного обходу дерева, починаючи з кореневого вузла і далі за гілками, поки не досягнеться листовий вузол.

## 2.5 Алгоритм побудови Chaid

CHAID (Chi-squared Automatic Interaction Detector) – це алгоритм для побудови дерева рішень, основною властивістю якого є обчислення хі-квадрату.

Значення хі-квадрат – це різниця між стандартним очікуваним сценарієм і фактичними результатами, які спостерігаються у даних.

Максимальне значення хі-квадрат є найбільш статистично значущим результатом у дереві рішень CHAID. Іншими словами, це найсильніший зв’язок між двома змінними зі знайдених значень хі-квадрат. Зазвичай алгоритм використовують для категоріальних даних.

Важливою особливістю є можливість множинних розбиттів, тобто дерево може мати більше двох вузлів.

Обчислюється хі-квадрат за формулою:

*.*

Тут:

* –спостережена частота (фактичні значення) для категорії;
* – очікувана частота для категорії;
* Σ – сума по всіх категоріях.

**Хід роботи алгоритму:**

1. Починаємо з ініціалізації кореневого вузла *CHAIDNode*, який містить вхідні дані і максимальну глибину дерева.
2. У вузлі Node обчислюється мітка *label* за допомогою методу get\_label(). Вона визначається як найбільш часто зустрічаюче значення класу в даних.
3. Для розбиття даних використовується метод *split().* Він шукає найкращу ознаку для розбиття завдяки функції *best\_split(),* використовуючи критерій χ² - *chi\_square\_test(col),* та перевіряє, чи розбиття є значущим за допомогою методу *critical\_value().* Якщо розбиття є значущим, вузол розбивається на піддерева.
4. Рекурсивний процес розбиття продовжується для кожного з дочірніх вузлів, доки не досягнуто максимальної глибини або розміру підгрупи, визначеної параметрами.
5. Ініціалізовуємо об'єкт *CHAID* з максимальною глибиною, яка передається при створенні.
6. Під час навчання метод *fit()* створює кореневий вузол дерева та викликає його метод *split()*, який рекурсивно розбиває вузли на піддерева.
7. Під час прогнозування метод *predict()* просто викликає метод *predict()* кореневого вузла для кожного вхідного рядка даних.

## 2.6 Алгоритм побудови Random Forest

Random Forest (випадковий ліс) – це алгоритм машинного навчання, який в процесі роботи формує сукупність дерев, а не одне дерево. Це і є його основною рисою та відмінністю. Назва має пряме відношення, оскільки ліс – це також множина хаотично створених дерев.

Алгоритм випадкового лісу поєднує результати кількох (випадково створених) дерев рішень для отримання кінцевого результату. Такий процес ще називають **Ensemble Learning** [6]. Зазвичай кожне дерево формується на базі алгоритму CART. Проте, можливо використовувати й інші, залежно від потреб і даних, з якими відбувається робота.

Кожне дерево в ансамблі працює із окремою підмножиною елементів. Формування цих підмножин відбувається завдяки методу беггінгу.

Беггінг (Bagging) – це процес, що означає формування випадкових підмножин із початкового тренувального набору. Пізніше відбувається зіставлення результатів кожного дерева для вибору найоптимальнішого та найточнішого результату. Це можна порівняти із «голосуванням», коли рішення вибирається на користь того, хто отримав найбільше голосів. Важливо зазначити, що елементи у підмножинах можуть бути обрані декілька разів, або можуть бути взагалі пропущені.

**Хід роботи алгоритму:**

1. Початкові налаштування:

Алгоритм починається з налаштування базових параметрів і створення необхідних класів. Для цього використовується клас Node, який представляє вузол дерева рішень.

1. Побудова окремого дерева рішень:

Для кожного дерева у випадковому лісі виконується процедура побудови дерева рішень за допомогою функції decision\_tree\_learning.

* Визначення результату вузла: Спочатку визначається, чи всі приклади мають однаковий результат. Якщо так, створюється листовий вузол з цим результатом.
* Вибір атрибуту для розбиття: Якщо результати різні, вибирається найкращий атрибут і порогове значення для розбиття за допомогою функції select\_best\_attribute. Розрахунок приросту інформації: Для кожного атрибуту розраховується приріст інформації за допомогою функцій information\_gain та entropy.
* Розділення даних: Дані розділяються на дві частини (ліву та праву) за допомогою функції split\_data.
* Рекурсивна побудова піддерев: Якщо можливе розбиття, рекурсивно будується ліве та праве піддерево.

1. Генерація випадкових підмножин даних:

Для побудови випадкового лісу створюється декілька дерев, кожне з яких тренується на випадковій підмножині вихідних даних. Це реалізовано у функції random\_forest\_learning.

* Побудова підмножин даних: Для кожного дерева випадковим чином вибирається підмножина даних з вихідного набору.
* Побудова дерев: На кожній підмножині даних будується окреме дерево рішень за допомогою функції decision\_tree\_learning.

1. Прогнозування за допомогою випадкового лісу:

Для прогнозування використовується функція random\_forest\_predict.

* Прогнозування окремим деревом: Для кожного прикладу прогнозуються результати кожного дерева в лісі за допомогою функції predict.
* Агрегація прогнозів: Остаточний прогноз визначається більшістю голосів серед усіх дерев.

# Розділ 3. Аналіз результатів

## 3.1 Програмна реалізація

Опис програми

Програма реалізує п'ять алгоритмів класифікації: ID3, C4.5, CART, CHAID та Random Forest. Вона приймає на вхід дані, обробляє їх та використовує для навчання моделей класифікації. Результати класифікації включають метрики точності та візуалізації, які допомагають проаналізувати ефективність кожного з алгоритмів для різних запитів.

Середовище розробки

Програмна реалізація проекту була здійснена на мові програмування Python у середовищі розробки Visual Studio. Python є зручною мовою для наукових обчислень та аналізу даних завдяки великій кількості доступних бібліотек.

Реалізація алгоритмів

Було реалізовано п'ять алгоритмів класифікації ID3, C4.5, CART, CHAID, Random Forest Для реалізації та тестування цих алгоритмів використовувались наступні бібліотеки:

* NumPy - основна бібліотека для наукових обчислень у Python. Вона забезпечує підтримку багатовимірних масивів та матриць, а також широкий набір математичних функцій для роботи з ними

Використання: для обробки числових даних, виконання математичних операцій та маніпуляцій з масивами.

* Pandas - бібліотека для аналізу та обробки даних. Вона надає структури даних, такі як DataFrame та Series, які дозволяють зручно та ефективно працювати з табличними даними.

Використання: для завантаження, обробки та аналізу даних, включаючи читання та запис даних у різних форматах (CSV, Excel), фільтрацію, групування та агрегування даних.

* Math - забезпечує набір математичних функцій, таких як обчислення тригонометричних, логарифмічних функцій та констант.

Використання: для виконання базових математичних операцій у реалізації алгоритмів.

* Time - забезпечує функції для роботи з часом, включаючи вимірювання часу, затримки та маніпуляції з датами.

Використання: для вимірювання часу виконання алгоритмів.

* Random - надає функції для генерації випадкових чисел та виконання різних випадкових операцій.

Використання: для генерації випадкових даних та виконання випадкових вибірок, що є важливим для деяких алгоритмів класифікації.

* Pydot- бібліотека для роботи з графами та їх візуалізації.

Використання: створення та візуалізації графічних представлень дерев рішень.

* Matplotlib - бібліотека для візуалізації даних, а Matplotlib.colors надає функції для роботи з кольорами в графіках.

Використання: Matplotlib використовувалась для створення графіків, діаграм та візуалізацій результатів класифікації, а саме для відображення результатів у вигляді Матриці Спряженості. Matplotlib.colors використовувалась для налаштування кольорових схем у графіках.

* Sys - забезпечує доступ до системних параметрів та функцій.

Використання: для обробки аргументів командного рядка та налаштувань середовища виконання програми.

* Sklearn.metrics - підмодуль бібліотеки scikit-learn, що надає різні метрики для оцінки якості моделей машинного навчання.

Використання: візуалізація ROC-кривої. Також варто відзначити, що ця бібліотеки має імплементовану частину даного дослідження. Оскільки можна застосувати функції для алгоритмів Cart та Random Forest. Проте інших алгоритмів знайдено не було. Також, проаналізувавши, застосування алгритмів у цій бібліотеці, було зроблено висновок про те, що дані не можливо передати у вигляді тренувальної та тестововї вибірки, а потрібно ділити на ознаки та цільові змінні, що додає певну складність у використанні.

Робота з даними та файлами

Робота з даними включала завантаження, попередню обробку, розподіл на навчальну та тестову вибірки, а також збереження результатів. Для цього використовувалась бібліотека pandas, що дозволяло ефективно читати дані з файлів CSV, очищувати та трансформувати їх перед подачею до алгоритмів класифікації. Дані використовувались переважно із платформи Kaggle. Перевага такого застосування це робота із реальними даними із різних сфер. Результати класифікації зберігались у вигляді таблиць та візуалізувались за допомогою matplotlib для подальшого аналізу.

Цей підхід забезпечив систематичний та гнучкий процес розробки та тестування алгоритмів класифікації, дозволяючи отримати детальні та візуально зрозумілі результати.

## 3.2 Вибірки даних

Для перевірки роботи алгоритмів, аналізу їхніх показників було використано вибірки із різних сфер, з різною кількість ознак та різною кількістю елементів.

Метою цього дослідження був пошук найоптимальніших застосувань алгоритмів класифікації, а саме: ID3, C4.5, CART, CHAID, Random Forest, для даних різного типу, щоб забезпечити різноманітні випадки, для точніших висновків.

Було обрано п'ять різних наборів даних, кожен з яких представляє характерний приклад застосування методів машинного навчання. Вибірки обиралися із різних сфер, що дає можливість продемонструвати актуальність теми для використання у різноманітних областях. Нижче наведено детальний опис кожного з використаних наборів даних:

1. Вибір, чи грати в теніс за даних погодних умов

Дана вибірка містить інформацію про погодні умови та рішення щодо гри в теніс. Основна мета цього набору даних – передбачити, чи буде проведено гру в теніс на основі зовнішніх факторів.

* Атрибути:
* Outlook (погода): Sunny, Overcast, Rainy;
* Temperature (температура): High, Mild, Cool;
* Humidity (вологість): High, Normal;
* Wind (вітер): Weak, Strong.
* Цільова змінна:
* PlayTennis: Yes, No.

Цей набір даних є класичним прикладом використання алгоритмів класифікації для прийняття рішень на основі простих факторів. Він використовується для перевірки, чи правильно будується модель.

1. Присвоєння стипендії студенту, зважаючи на його бали, активність і відвідуваність.

У цих даних міститься інформація про студентів та їхні академічні показники, активність та відвідуваність. Метою є запропонувати такий програмний спосіб оцінювання чи варто надавати студентові стипендію.

* Атрибути:
* Marks (бали): від 0 до 100, у вигляді A–F системи для трьох предметів;
* Activity (активність): оцінка від 0 до 5 – показник додаткових балів;
* Attendance\_grade (відвідуваність): yes/no – допуск по відвідуваності.
* Цільова змінна:
* Scholarship: Yes, No.

Такий підхід допомагає зрозуміти, як різні фактори впливають на рішення щодо присвоєння стипендій. Також, це дає можливість будувати рейтингові списки враховуючи більшу кількість показників, а не тільки середній бал.

1. Прогноз розлучення на основі питань про шлюбне життя.

Зазначена вибірка містить інформацію про шлюбне життя пар, включаючи відповіді на низку запитань, що стосуються їхніх стосунків. Метою є передбачити, чи відбудеться розлучення. Інформація була опублікована на платформі Kaggle [!]

* Атрибути:
* 54 питання, які стосуються подружжя та їхніх взаємин, на яке дають відповіді у вигляді:

(0=Never, 1=Seldom, 2=Averagely, 3=Frequently, 4=Always).

* Цільова змінна:
* Divorce: Yes, No.

Цей набір даних надає можливість вивчити взаємозв'язок між різними аспектами шлюбних стосунків і їхнім кінцевим результатом та виокремити, які аспекти стають найвагомішими. Таке дослідження може застосовуватися для психологів та їхніх спостережень.

1. Прогноз майбутнього у компанії співробітників залежно від отриманої інформації.

Це сукупність даних, яка надано HR-відділом про співробітників. Вона містить деталі про співробітників компанії, включаючи їхню освіту, історію роботи, демографічні дані та фактори, пов’язані з роботою. Метою є передбачити, чи залишиться співробітник в компанії.

* Атрибути (визначено 8 різних характеристик):
* Education (освіта): Bachelor's, Master's, PhD;
* WorkHistory (історія роботи): Joining Year (Рік приєднання), Experience in Current Domain (Досвід в поточній галузі);
* Demographics (демографічні дані): Age (вік), Gender (стать), City (місто);
* JobFactors (фактори, пов'язані з роботою): Payment Tier (Рівень оплати), Ever Benched (Чи був на бенчі – тимчасово без зайнятості на проекті).
* Цільова змінна:
* Leave or Not (Чи покинув): Це цільовий стовпчик, який вказує, чи залишив співробітник компанію або ні.

Така інформація допомагає зрозуміти, які фактори впливають на утримання співробітників у компанії. Також це можливість до оптимізація роботи HR-відділу.

1. Прогноз задоволеності клієнтів авіакомпанії на основі різних параметрів.

Дана сукупність даних містить інформацію про рівень задоволеності клієнтів у певній (не зазначеній) авіакомпанії. Метою є передбачити, чи будуть майбутні клієнти задоволені на основі різних параметрів, включених у вибірку.

* Атрибути (Загальна кількість ознак – 22 штуки):
* Деталі про клієнта (Тип клієнта, Вік);
* Інформація про подорож (Тип подорожі, Відстань, Клас, Затримка у відправленні чи прибутті);
* Перелік пунктів для оцінки послуг (Розташування, складений розклад, харчування та інші додаткові послуги);
* Перелік пунктів для оцінки зручностей (Онлайн-підтримка, обслуговування, користування сервісом бронювання, багаж, посадка та інші).
* Цільова змінна:
* CustomerSatisfaction (Задоволення якістю): Satisfied, Dissatisfied.

Це дає нам краще розуміння про фактори, що впливають на задоволеність клієнтів авіакомпанії, та допомагає розробляти стратегії для покращення обслуговування.

## 3.3 Аналіз метрик

У даному розділі було проведено порівняльний аналіз ефективності алгоритмів класифікації ID3, C4.5, CART, CHAID та Random Forest на різних вибірках даних. Для кожної вибірки було обчислено такі основні метрики:

* час виконання;
* точність;
* чутливість;
* правильність;
* специфічність;
* F1-міра.

Кожна метрика дозволяє оцінити різні аспекти роботи алгоритмів, такі як здатність правильно класифікувати дані, обробляти великі обсяги даних швидко та ефективно, а також виявляти різні типи помилок класифікації.

Вибірки даних, використані в аналізі, охоплюють різні області застосування, що дозволяє оцінити універсальність та надійність алгоритмів у різних контекстах. Зокрема, було досліджено вибірки, пов'язані з прийняттям рішень на основі погодних умов, присвоєнням стипендій студентам, прогнозуванням розлучень, утриманням співробітників у компаніях та оцінкою задоволеності клієнтів авіакомпаній (детальніше – див розділ 3.1).

Нижче показані таблиці з результатами обчислень для кожної вибірки, що ілюструють ефективність кожного алгоритму за вказаними метриками. Ці результати допоможуть зробити висновки щодо доцільності використання кожного з алгоритмів для різних типів задач класифікації та прогнозування.

Першим показником було взято час роботи алгоритму (а саме, побудови дерева), одиниця виміру якого – секунда. Цей показник є дуже суттєвим, оскільки часто це прямо пов’язано із ресурсами, необхідними для роботи, що впливає на витрати для апаратного забезпечення.

У лівому стовпці подано інформацію про кількість елементів у вибірці у форматі: тренувальна вибірка/тестова вибірка. Зверху таблиці перераховано алгоритми, на яких відбувалось тестування. Нижче показано таблицю із вказаним часом роботи для кожного алгоритму:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| 20/5 | 0.508006 | 0.01686978 | 0.01834154 | 0.028433561 | 0.10077023 |
| 100/20 | 0.1613023 | 0.024705171 | 0.03152513 | 0.512581586 | 0.03034782 |
| 800/100 | 0.567224 | 0.04337501 | 0.43833518 | 2.766915082 | 0.15512204 |
| 4000/600 | 12.505325 | 1.21067571 | 77.9678659 | 21.0148625 | 7.14400577 |
| 20к/4к | 104.65811 | 193.297098 | 5759.99492 | 963.22899 | 1276.63264 |

Час, секунда

Табл. 1. Назва

Проаналізувавши отримані результати, можна відзначити, що на маленьких вибірках даних найшвидше працює CART, проте на великих вибірках його швидкодія є найгіршою. На великих вибірках суттєво швидшими є C45 та ID3.

Наступним етапом було визначення точності роботи алгоритмів (посилання на ф-лу). Це найбільш узагальнений показник, який дає інформацію про коректність роботи побудованої моделі. Результати подано нижче:

Точність

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| 20/5 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 100/20 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 800/100 | 0.75 | 0.85 | 0.85 | 1.0 | 0.9 |
| 4000/600 | 0.5770992 | 0.796946564 | 0.8045802 | 0.8167938 | 0.8174809 |
| 20к/4к | 0.44775 | 0.9035 | 0.923 | 0.77725 | 0.9375 |

Табл. 2. Назва

Кожен із алгоритмів на вибірках із малою кількістю даних демонструє однозначну точність. Проте, зі зростанням даних результати розбігаються. Random Forest має найвищу точність. Менший, але все ще високий відсоток точності зберігають алгоритми CART, Chaid і C45.

Під час перевірки роботи алгоритмів було обчислено ще такі показники:

* чутливість – важлива в ситуаціях, коли важливо знайти всі позитивні випадки;
* правильність – необхідна, коли треба мінімізувати хибно позитивні випадки;
* специфічність – суттєва, коли важливо правильно виявляти негативні випадки;
* F1-оцінка – корисна, коли потрібно знайти баланс між чутливістю і правильністю. Вона особливо корисна у випадках з незбалансованими даними.

**150/20**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| Чутливість |  |  |  |  |  |
| Правильність |  |  |  |  |  |
| Специфічність |  |  |  |  |  |
| F1-оцінка |  |  |  |  |  |

Табл. 3. Назва

…

**4000/600 –** вибірка про працевлаштування

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| Чутливість | 0.52941176 | 0.638009049 | 0.660633484 | 0.59728506 | 0.660633484 |
| Правильність | 0.78 | 0.726804123 | 0.733668341 | 0.80981595 | 0.753478260 |
| Специфічність | 0.92396313 | 0.877880184 | 0.877880184 | 0.9285714 | 0.882442396 |
| F1-оцінка | 0.63072776 | 0.679518072 | 0.695238095 | 0.68750001 | 0.700987654 |

Табл. 4. Назва

Для даної вибірки даних можна зробити такі висновки: чутливість та F1-оцінка мають найбільший показник у CART та Forest. Правильність та специфічність є найвищими у Chaid. Зважаючи на інтерпретацію вибірки, варто звернути увагу на такі показники, як чутливість і специфічність. Оскільки, важливо вчасно зауважити намір покинути компанію у цінного співробітника.

**20 000/4 000 –** вибірка про оцінку якості роботи в авіакомпанії

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| Чутливість | 0.4878048 | 0.91302346 | 0.93511274 | 0.75563736 | 0.945697 |
| Правильність | 0.80181543 | 0.9096744 | 0.92405638 | 0.82017982 | 0.9487534 |
| Специфічність | 0.85659551 | 0.89217296 | 0.90859332 | 0.80295566 | 0.9392446 |
| F1-оцінка | 0.60658082 | 0.9113458 | 0.92955169 | 0.78658682 | 0.9472228 |

Табл. 5. Назва

**…..**

**….**

**….**

**….**

Окремо було досліджено вплив обсягу набору даних на час і точність роботи алгоритмів на прикладі однієї вибірки – набору даних із оцінкою якості роботи авіакомпанії Результат такого спостереження дасть інформацію про масштабованість алгоритму.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| 50/10 | 0.21571 | 0.08803 | 0.028800 | 5.1147391 | 0.0901050 |
| 500/100 | 1.722869 | 0.971092 | 2.762029 | 44.3249108 | 3.0164046 |
| 5000/1000 | 37.05520 | 22.156417 | 313.622 | 302.17766 | 130.0449 |
| 50000/1000 | 348.01483 | 722.56463 | 15год+ | 15+ | 10680.057 |

Час

Табл. 6. Назва

У кожному алгоритмі відбувається різке зростання часу виконання. Проте, пропорційна залежність не простежується.

Точність

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ID3 | C45 | CART | CHAID | Forest |
| 50/10 | 0.0 | 0.7 | 0.8 | 0.4 | 0.7 |
| 500/100 | 0.06 | 0.8 | 0.78 | 0.52 | 0.83 |
| 5000/1000 | 0.259 | 0.873 | 0.883 | 0.551 | 0.918 |
| 50000/1000 | 0.5021 | 0.921 | - | - | 0.946 |

Табл. 7. Назва

Внаслідок отриманих результатів можна легко переконатися, що зі збільшенням кількості даних для тренування вибірки, зростає точність результату під час перевірки роботи моделі на тестових даних.

## 3.4 Візуалізація продуктивності роботи алгоритмів за допомогою Confusion Matrix

Для наочної оцінки продуктивності класифікаційних моделей, було створено матриці спряженості для кожного алгоритму на різних наборах даних. Матриця спряженості наочно демонструє кількість правильних і неправильних передбачень, що дозволяє оцінити якість моделі та ідентифікувати типи помилок. Нижче показано скріншоти результатів програми у вигляді матриць спряженості.

|  |  |
| --- | --- |
| Рис.3. Матриця ID3 | Рис.4. Матриця С45 |
| Рис.5. Матриця CART | Рис.6. Матриця Chaid |
| Рис.7. Матриця Random Forest | |

***Коментар***

## 3.5 Візуалізація продуктивності роботи алгоритмів за допомогою ROC-кривої

Для додаткової оцінки продуктивності класифікаційних моделей також була побудована крива ROC (Receiver Operating Characteristic). Ця крива відображає співвідношення між чутливістю та специфічністю моделі при різних порогових значеннях. Вона є важливим інструментом для визначення оптимального порогу класифікації та порівняння різних моделей за їхньою здатністю правильно класифікувати дані. Нижче показані скріншоти результатів програми у вигляді ROC-кривої.

|  |  |
| --- | --- |
| Рис.8. ROC-крива ID3 | Рис.9. ROC-крива C45 |
| Рис.10. ROC-крива Cart | Рис.11. ROC-крива Chaid |
| Рис.12. ROC-крива Random Forest | |

Результати ROC кривої особливі тим, що вони дозволяють оцінити ефективність класифікаційної моделі при різних порогових значеннях. Ця крива відображає залежність між чутливістю (true positive rate) та специфічністю (true negative rate) моделі. Чим ближче крива до верхнього лівого кута графіка, точки з координатами (0;1), тим краще модель може розрізняти позитивні і негативні класами при різних порогових значеннях.

Особливість результату полягає в тому, що крива ROC дає можливість визначити оптимальний поріг класифікації для моделі, що максимізує чутливість і специфічність одночасно.

***Коментар***

# **ВИСНОВКИ**

У бакалаврській роботі вирішене поставлене завдання, яке полягало у дослідження оптимальності алгоритмів дерева рішень для виокремлення їхнього найдоцільнішого застосування. Отримані наступні результати:

1. виокремлено критерії оцінки оптимальності роботи алгоритмів;
2. на основі цих критеріїв отримано метрики для оцінки оптимальності роботи алгоритмів;
3. з використанням метрик здійснено порівняльний аналіз оптимальності роботи алгоритмів;
4. окреслені межі застосувань алгоритмів.

**Опис отриманих результатів**

В ході дослідження було проведено порівняльний аналіз ефективності різних алгоритмів класифікації на різних типах даних. Результати дослідження показали, що алгоритми, такі як C45 та CART, проявили себе ефективними на их вибірках з числовими атрибутами завдяки формуванню порогових значень. Стосовно категоріальних даних, алгоритми CART та Random Forest продемонстрували найкращі результати, оскільки вони ефективно працюють з категоріальними змінними та розпізнають складні взаємозв'язки між ними.

Якщо зважати на показник часу, тоді варто застосовувати C45 або ID3. Проте задовільні результати ID3 буде давати лише на невеликих та простих вибірках, оскільки на складних демонструє низьку точність. На великих обсягах даних алгоритм C45 показав себе краще, оскільки він володіє високою швидкістю обробки та може ефективно працювати з великим числом атрибутів, забезпечуючи при цьому високу точність класифікації.

Щодо результатів метрик, можна відзначити, що алгоритми C45 та Random Forest демонстрували високу точність та чутливість у класифікації, що підтверджує доцільність використання у вибірках, де критично важливо правильно визначати позитивні елементи. Зокрема, це важливо у медичній сфері, де завданням є визначення всіх випадків діагнозів.

Показник правильності має високі результати у алгоритмі Chaid, тобто це кількість правильно визначених елементів серед усіх даних. Тобто, якщо брати як приклад, опрацьовані вибірки, то правильне присвоєння стипендії буде мати переважаючу мету.

Наступна метрика для аналізу – це специфічність, відсоток коректно визначених негативних відповідей. Такі результати часто застосовують для класифікацій, що спрямовані на пошук негативних відповідей. Зокрема, при дослідженні відгуків про сервіс компанії більше зважають на негативні відгуки.

Останнім етапом аналізу була F-міра, яка є балансом між чутливістю та правильністю, враховуючи якості обидвох метрик. Вона буде більш вживаною у даних, де є переважання одного із класів. Оскільки, якщо порівнювати із точністю, яка може давати високу оцінку правильно прокласифікованих результатів, не зважаючи на те, що клас із меншою розмірністю може мати велику похибку.

**Апробація результатів бакалаврської роботи.** Основні результати бакалаврської роботи доповідалися на Міжнародній студентській науковій конференції з прикладної математики та комп’ютерних наук МСНКПМКН – 2024 (Львів, 16-17 травня 2024 р.). Також була зроблена публікація у тезах доповідей [Христина Бень Галина Квасниця. Порівняння оптимальності алгоритмів класифікації даних на прикладі дерев рішень. *Міжнародна студентська наукова конференція з прикладної математики та комп’ютерних наук МСНКПМКН – 2024*.

**Перспектива розширення дослідження**

Зважаючи на кількість застосувань даного алгоритму у різних сферах перспектива продовження дослідження є беззаперечною. Можливістю для розвитку роботи може бути розширення обсягу дослідження на інші типи даних, такі як текстові або зображення, а також удосконалення алгоритмів для досягнення ще вищих результатів у класифікації. Також може бути застосовано інший підхід для використання отриманих результатів: створення користувацької платформи, яка буде вирішувати питання класифікації та прогнозування у загальному доступі. Ще один спосіб для розвитку даного результату – це створення бібліотеки у Python, у якій можна буде використовувати різні алгоритми та допоміжні функції для роботи з даними.

# Джерела

Матеріали конференції. – Львів, 16-17 травня 2024 року. – с. 204-207. – Режим доступу: <https://ami.lnu.edu.ua/wp-content/uploads/2024/05/ISSCAMCS-2024.pdf>].

1. Задачі Data Mining. Класифікація та кластеризація [Електронний ресурс] Режим доступу до ресурсу: https://moodle.znu.edu.ua/pluginfile.php /486129/mod\_resource/content/1/%D0%9B%D0%B5%D0%BA%D1%86%D1%96%D1%8F%206.pdf
2. Інформаційні системи та технології в управлінні. [Електронний ресурс] Режим доступу до ресурсу: http://eir.zntu.edu.ua/bitstream/123456789/342/ 1/met\_vk\_bila\_3.pdf
3. Класифікація алгоритмів машинного навчання: лінійна регресія, класифікація та кластеризація [Електронний ресурс] Режим доступу до ресурсу: https://bloginnovazione.it/uk/machine-learning/3716/
4. Рівняня регресії У на Х та Х на У. Алгоритм та побудова [Електронний ресурс] Режим доступу до ресурсу: https://yukhym.com/uk/vipadkovi-velichini/rivniania-rehresii-alhorytm-ta-pobudova.html
5. Логістична регресія[Електронний ресурс] Режим доступу до ресурсу: http://om.univ.kiev.ua/users\_upload/15/upload/file/lecture\_sm\_08.pdf
6. Огляд логістичної регресії [Електронний ресурс] Режим доступу до ресурсу: https://zephyrnet.com/uk/%D0%BE%D0%B3%D0%BB%D1%8F%D0%B4-%D0%BB%D0%BE%D0%B3%D1%96%D1%81%D1%82%D0%B8%D1%87%D0%BD%D0%BE%D1%97-%D1%80%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%96%D1%97/
7. How to Use ROC Curves and Precision-Recall Curves for Classification in Python [Електронний ресурс] Режим доступу до ресурсу: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/
8. Confusion Matrix for Your Multi-Class Machine Learning Model [Електронний ресурс] Режим доступу до ресурсу: https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826
9. Методи дерев рішень, класифікації та прогнозування [Електронний ресурс] Режим доступу до ресурсу: https://moodle.znu.edu.ua/pluginfile.php?file=/486136 /mod\_resource/content/1/%D0%9B%D0%B5%D0%BA%D1%86%D1%96%D1%8F%209.pdf
10. Дерева рішень і алгоритми їх побудови [Електронний ресурс] Режим доступу до ресурсу: https://ekmair.ukma.edu.ua/server/api/core/bitstreams/590b3734-7e5c-48cc-80ec-f1704589696b/content