Національний технічний університет України «КПІ ім. Ігоря Сікорського» Факультет Інформатики та Обчислювальної Техніки

Кафедра інформаційних систем та технологій

Лабораторна робота №3

з дисципліни «Інтелектуальний Аналіз Даних»

на тему

«ДЕРЕВА ВИРІШАЛЬНИХ ПРАВИЛ»

Варіант №12

Виконав:

студент групи ІС-02

Плостак І. М.

Київ – 2021

**1. Мета роботи**

Вивчити основні методи побудови дерев вирішальних правил, навчитися використовувати спеціалізовані програмні засоби для побудови дерев вирішальних правил.

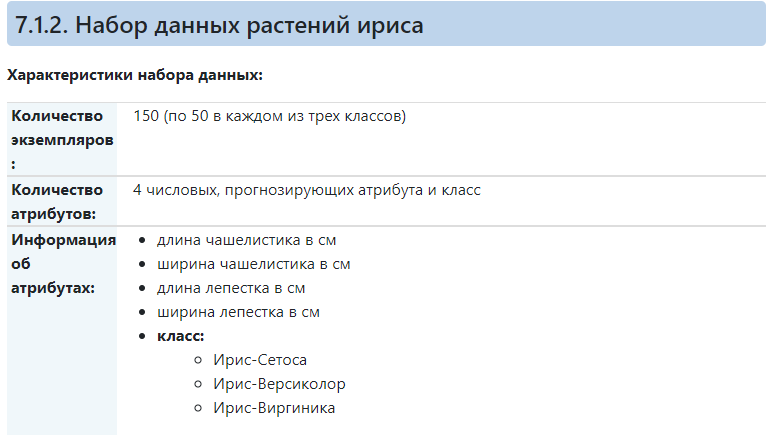
**2. Завдання до роботи**

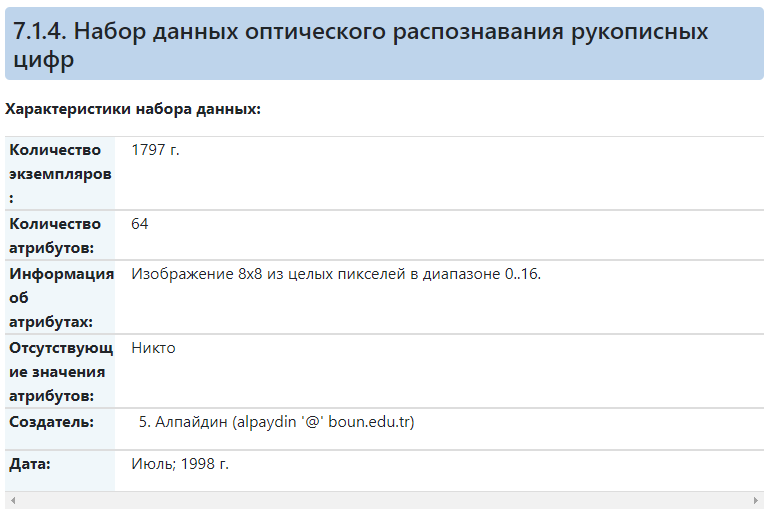
1. Ознайомитися з конспектом лекцій та рекомендованою літературою, а також додатком Б, що містить опис програмного забезпечення для побудови дерев вирішальних правил.
2. Сформувати набір даних для обробки та аналізу.
3. Використовуючи рекомендоване програмне забезпечення здійснити обробку набору даних з метою побудови дерева вирішальних правил.
4. Використати побудоване дерево для прийняття рішень на конкретному прикладі.
5. Оформити звіт з роботи.
6. Відповісти на контрольні питання.

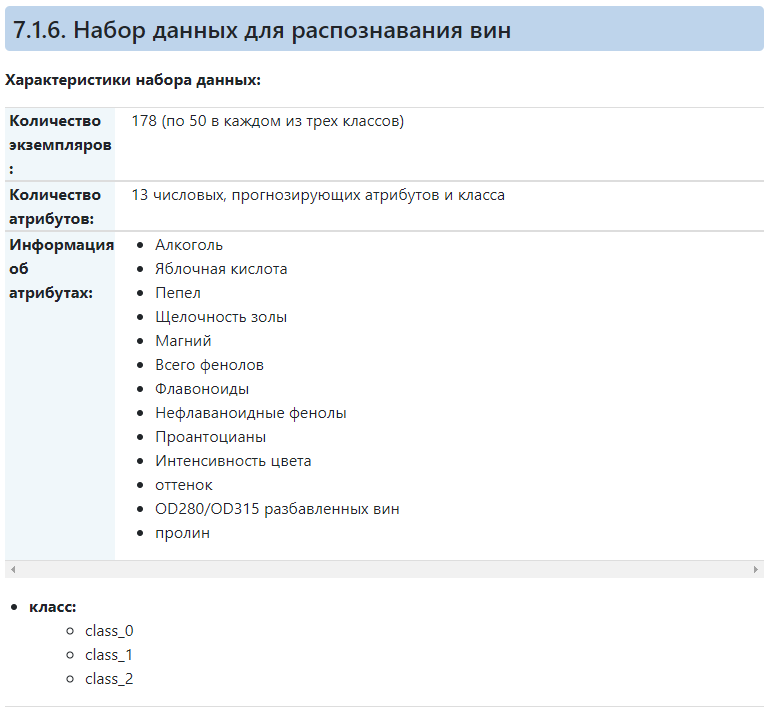
**3. Набір даних для обробки**

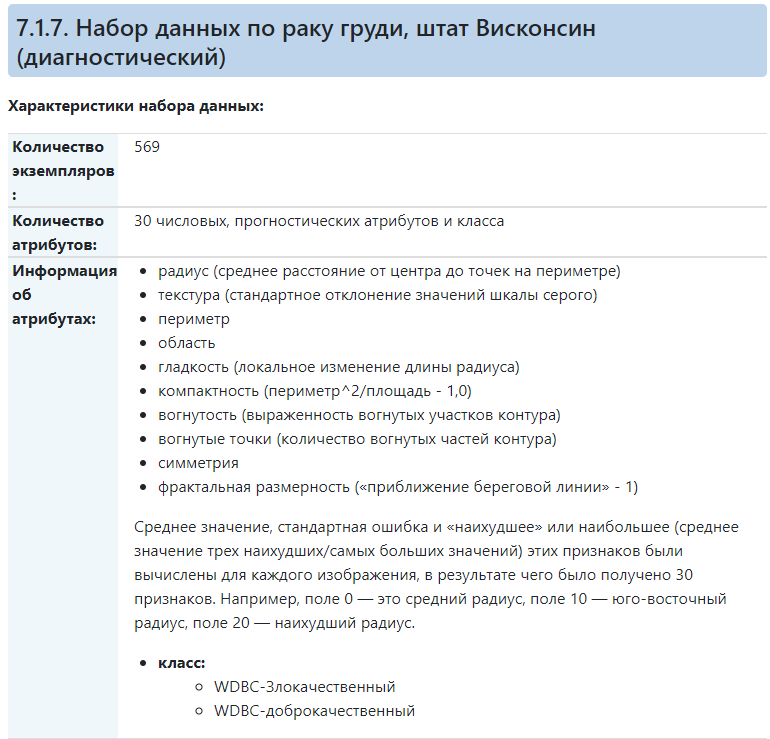
Для обробки було взято чотири різних набори даних:

Iris, Digits, Wine, Breast\_Cancer – чотири вбудовані набори даних у бібліотеку sklearn.









**4. Лістинг основних функцій програми**

main.py:

import numpy as np  
from sklearn.tree import DecisionTreeClassifier, export\_graphviz, plot\_tree  
from sklearn.model\_selection import cross\_val\_score, train\_test\_split  
from sklearn.datasets import load\_iris, load\_digits, load\_wine, load\_breast\_cancer  
from sklearn import metrics, tree  
from six import StringIO  
from IPython.display import Image, display  
import pydotplus  
import matplotlib.pyplot as plt  
import matplotlib.image as img  
  
import funcsForTrees as fc  
  
set = load\_iris() # load\_iris, load\_digits, load\_wine, load\_breast\_cancer  
X = set.data  
y = set.target  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=0)  
  
tree\_model = DecisionTreeClassifier()  
tree\_model.fit(X\_train, y\_train)  
  
dot\_data = StringIO()  
export\_graphviz(tree\_model,  
 out\_file=dot\_data,  
 filled=True,  
 rounded=True,  
 special\_characters=True,  
 feature\_names=set.feature\_names)  
graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())  
graph.write\_png('modelTree.png')  
Image(graph.create\_png())  
tree.plot\_tree(tree\_model, impurity=False, filled=True, feature\_names=set.feature\_names)  
plt.show()  
  
y\_pred = tree\_model.predict(X\_test)  
print("Точність початкового дерева:")  
print(metrics.accuracy\_score(y\_test, y\_pred))  
  
indexOfBest, ccp\_alphas, res1, res2 = fc.searchCcpAlpha(X\_train, y\_train, X\_test, y\_test, tree\_model)  
print('-'\*40)  
print("Вибране {0} ccp\_alpha зі значенням = {1}.\n"  
 "Точність тренувального набору даних: {2}.\n"  
 "Точність тестувального набору даних: {3}.".format(indexOfBest,  
 ccp\_alphas[indexOfBest],  
 res1,  
 res2))  
print('-'\*40)  
  
tree\_model = DecisionTreeClassifier(ccp\_alpha=ccp\_alphas[indexOfBest])  
tree\_model.fit(X\_train, y\_train)  
  
dot\_data = StringIO()  
export\_graphviz(tree\_model,  
 out\_file=dot\_data,  
 filled=True,  
 rounded=True,  
 special\_characters=True,  
 feature\_names=set.feature\_names)  
graph = pydotplus.graph\_from\_dot\_data(dot\_data.getvalue())  
graph.write\_png('modelOptimTree.png')  
Image(graph.create\_png())  
tree.plot\_tree(tree\_model, impurity=False, filled=True, feature\_names=set.feature\_names)  
plt.show()  
  
y\_pred = tree\_model.predict(X\_test)  
print("Точність оптимізованого дерева:")  
print(metrics.accuracy\_score(y\_test, y\_pred))

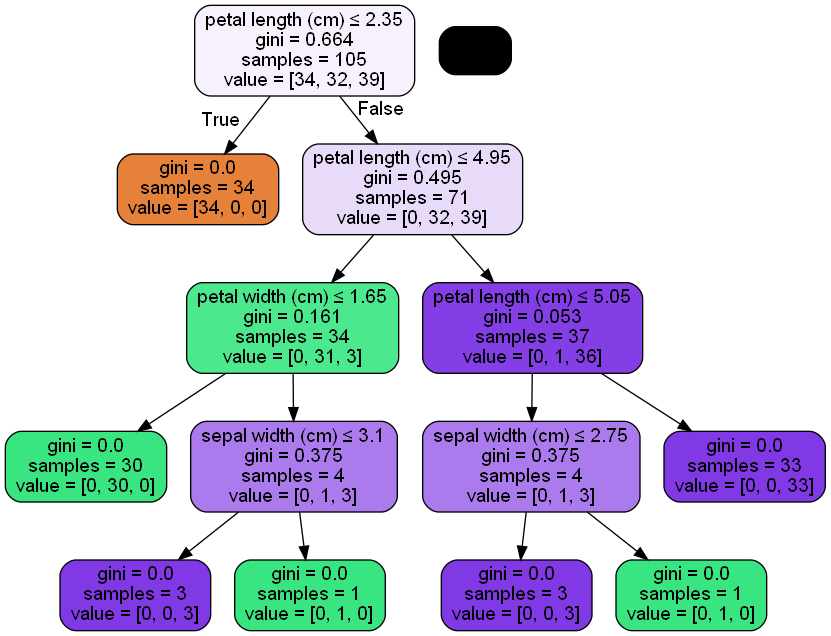
funcsForTrees.py:

from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import cross\_val\_score, train\_test\_split  
from matplotlib import pyplot as plt  
  
def searchInAplhas(ccp\_alphas, tests, trains):  
 indexOfBest = 0  
 for i in range(len(ccp\_alphas)):  
 if trains[i] >= tests[i]:  
 if tests[i] > tests[indexOfBest]:  
 indexOfBest = i  
 elif tests[i] == tests[indexOfBest]:  
 if trains[i] > trains[indexOfBest]:  
 indexOfBest = i  
 elif trains[i] == trains[indexOfBest]:  
 if ccp\_alphas[i] > ccp\_alphas[indexOfBest]:  
 indexOfBest = i  
 else:  
 if trains[i] > trains[indexOfBest]:  
 indexOfBest = i  
 elif trains[i] == trains[indexOfBest]:  
 if tests[i] > tests[indexOfBest]:  
 indexOfBest = i  
 elif tests[i] == tests[indexOfBest]:  
 if ccp\_alphas[i] > ccp\_alphas[indexOfBest]:  
 indexOfBest = i  
 return indexOfBest, ccp\_alphas, tests[indexOfBest], trains[indexOfBest]  
  
def searchCcpAlpha(X\_train, y\_train, X\_test, y\_test, tree):  
 path = tree.cost\_complexity\_pruning\_path(X\_train, y\_train)  
 ccp\_alphas = path.ccp\_alphas  
 ccp\_alphas = ccp\_alphas[:-1]  
  
 trees = []  
 for ccp\_alpha in ccp\_alphas:  
 temp\_tree = DecisionTreeClassifier(ccp\_alpha=ccp\_alpha)  
 temp\_tree.fit(X\_train, y\_train)  
 trees.append(temp\_tree)  
  
 train\_scores = [derevo.score(X\_train, y\_train) for derevo in trees]  
 test\_scores = [derevo.score(X\_test, y\_test) for derevo in trees]  
  
 # Uncomment to draw a graph of ccp\_alphas  
 #'''  
 fig, ax = plt.subplots()  
 ax.set\_xlabel("alpha")  
 ax.set\_ylabel("accuracy")  
 ax.set\_title("Accuracy vs alpha for training and testing sets")  
 ax.plot(ccp\_alphas, train\_scores, marker="o", label="train", drawstyle="steps-post")  
 ax.plot(ccp\_alphas, test\_scores, marker="o", label="test", drawstyle="steps-post")  
 ax.legend()  
 plt.show()  
 #'''  
  
 return searchInAplhas(ccp\_alphas, test\_scores, train\_scores)

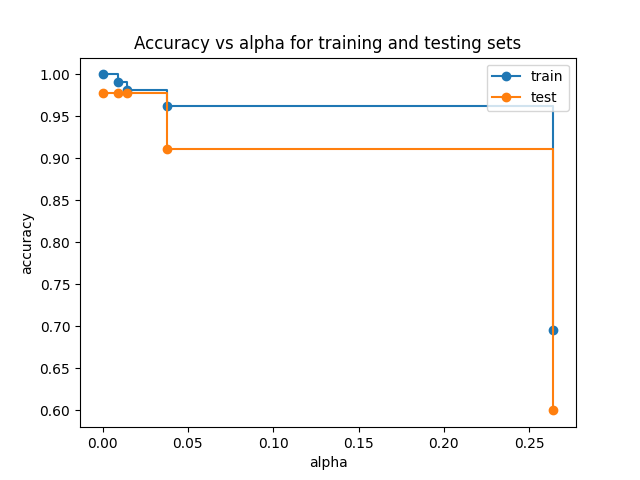
**5. Результати роботи програмного забезпечення**

**Набір даних Iris:**

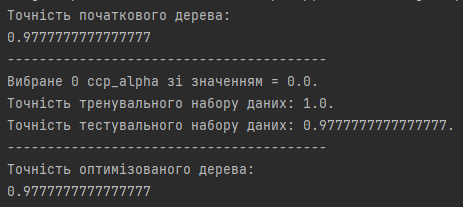
Початкове дерево:



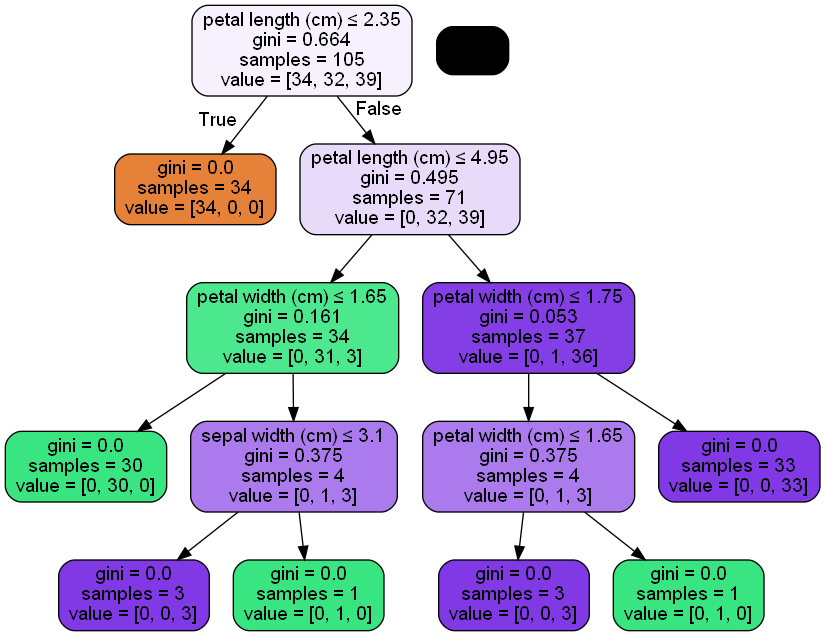
Графік вартісних параметрів складності залежно від точності дерев:



Вивід програми:

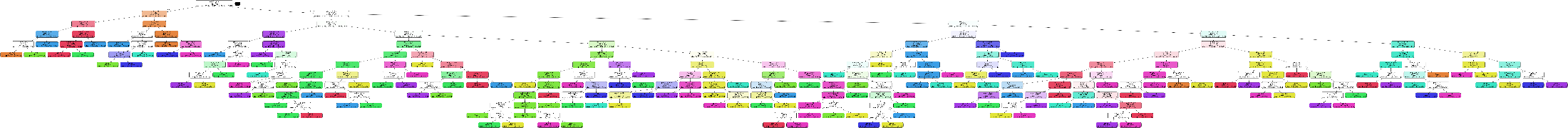


Оптимізоване дерево:

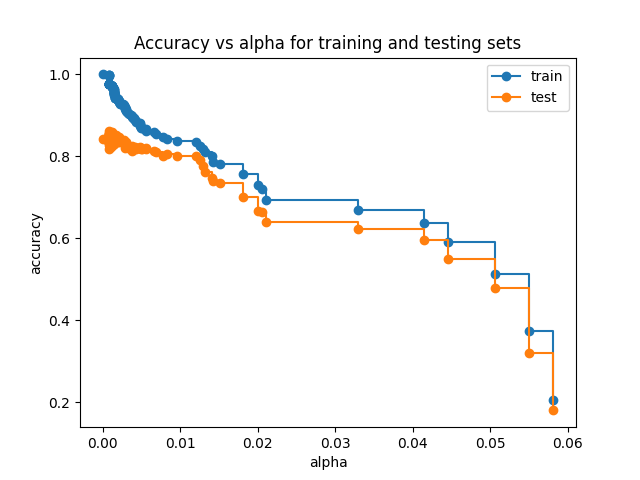


**Набір даних Digits:**

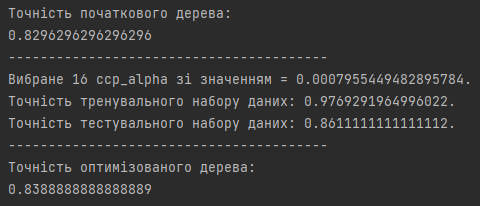
Початкове дерево:



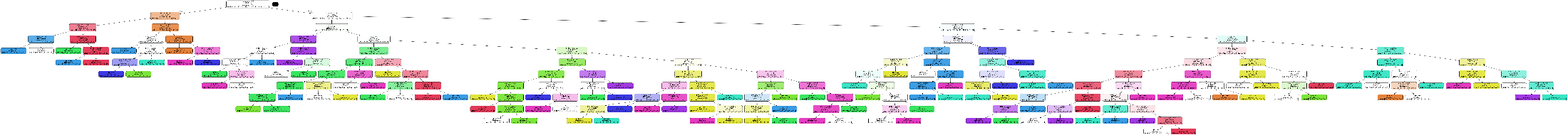
Графік вартісних параметрів складності залежно від точності дерев:



Вивід програми:



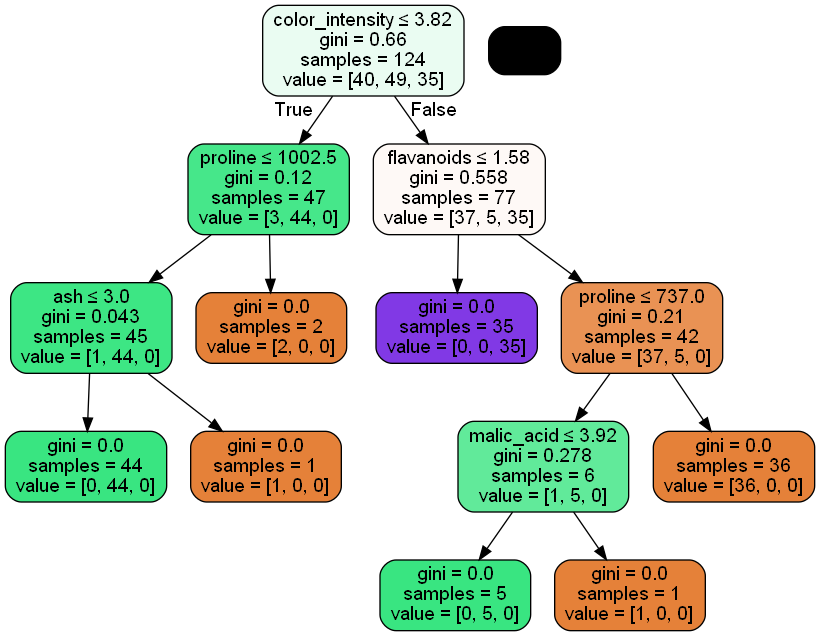
Оптимізоване дерево:



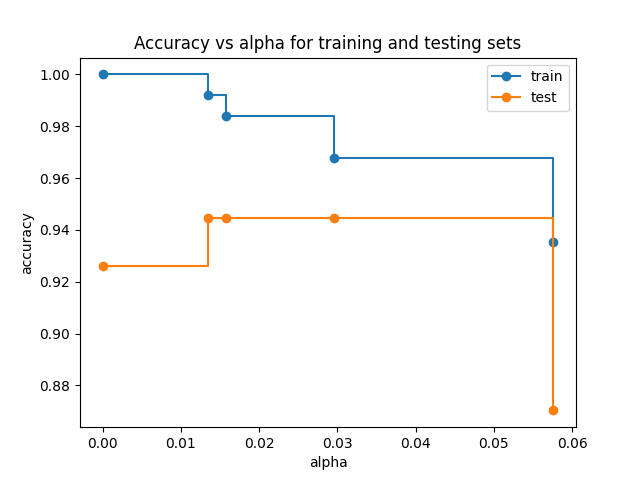
Приклад ДУЖЕ великого набору даних, але навіть тут вдалося покращити точність дерева.

**Набір даних Wine:**

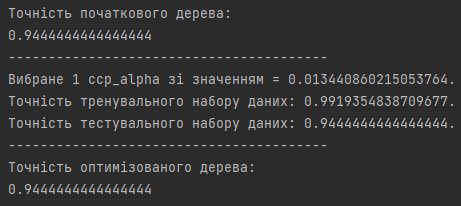
Початкове дерево:



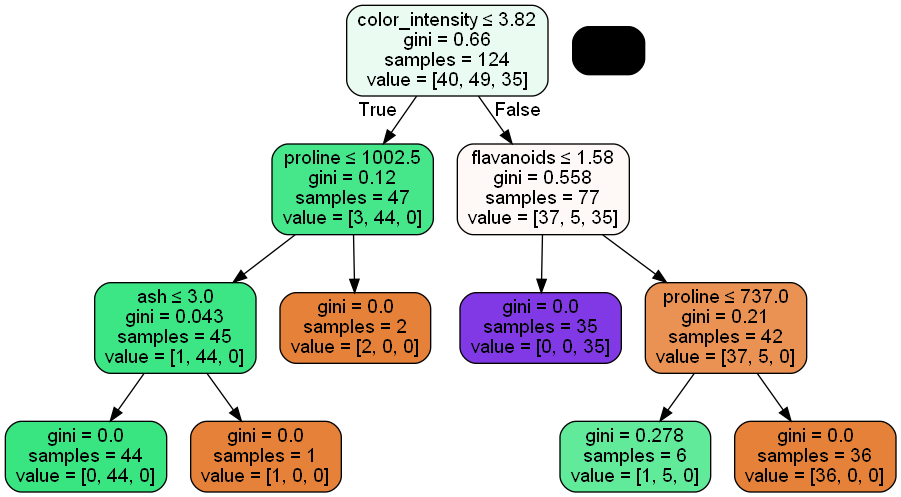
Графік вартісних параметрів складності залежно від точності дерев:



Вивід програми:

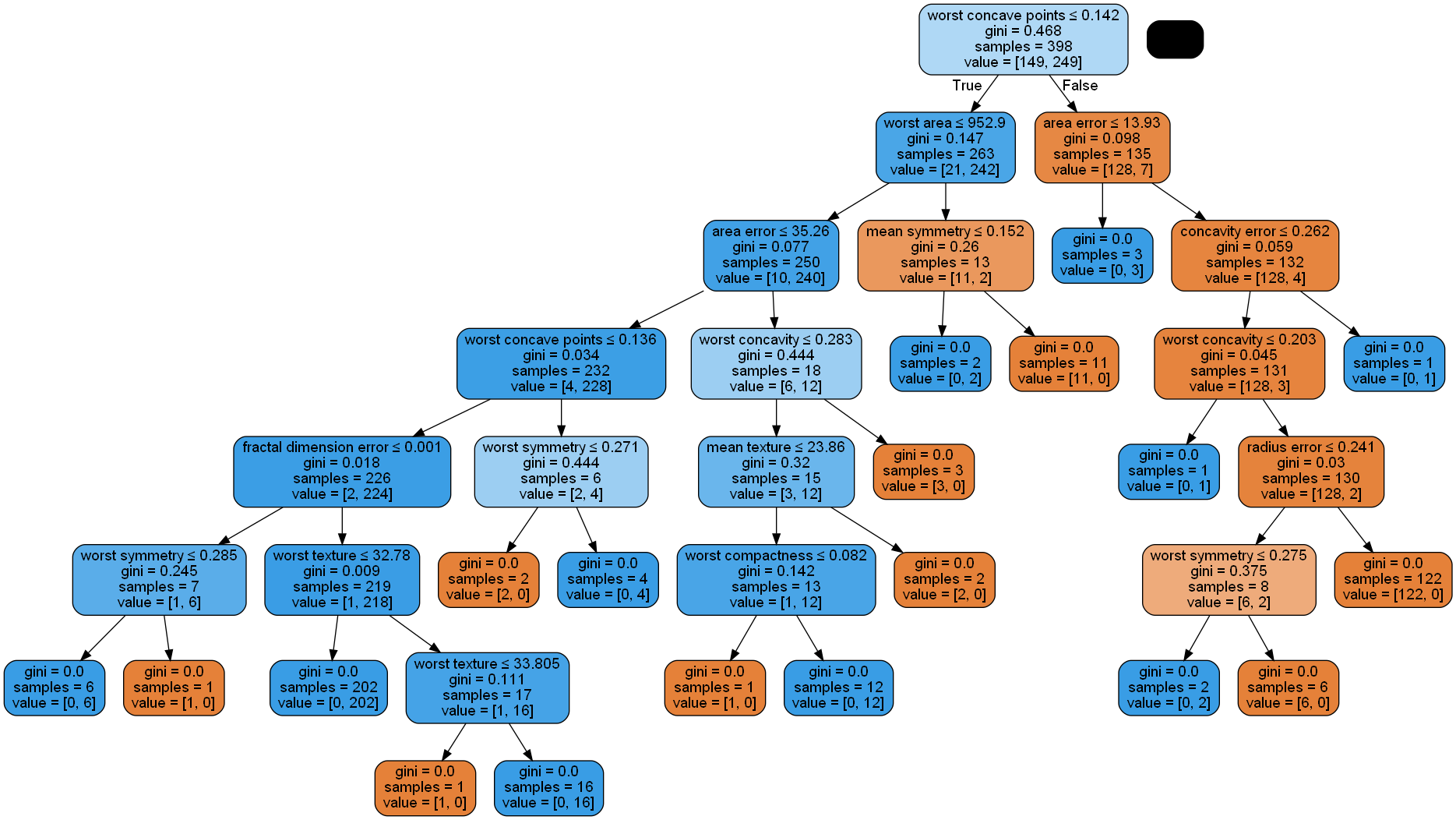


Оптимізоване дерево:

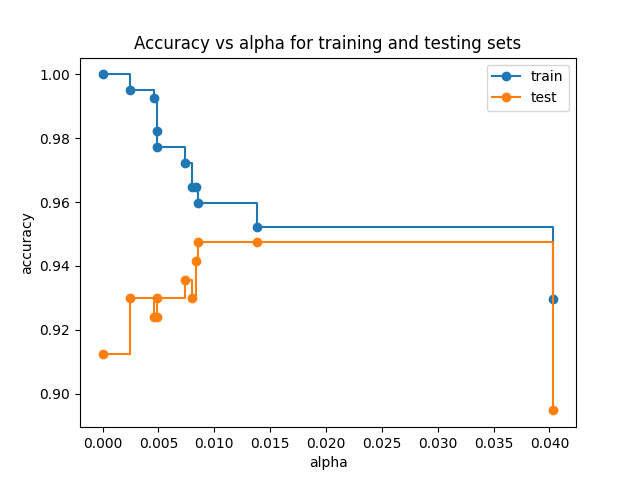


**Набір даних Breast Cancer:**

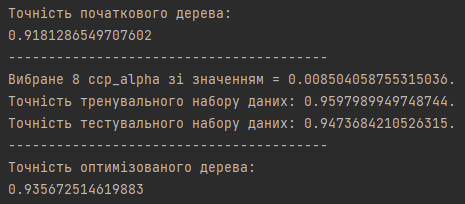
Початкове дерево:



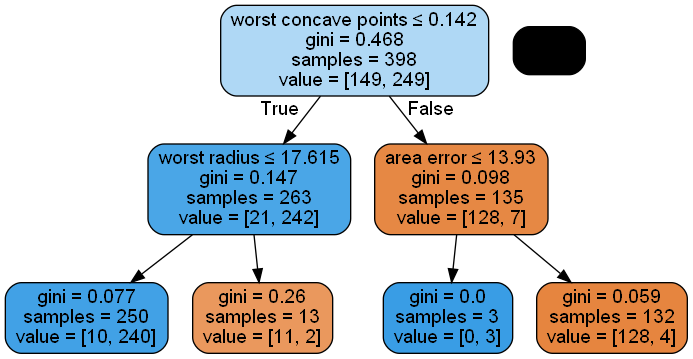
Графік вартісних параметрів складності залежно від точності дерев:



Вивід програми:



Оптимізоване дерево:



**6.1. Контрольні питання**

1. Що таке дерево вирішальних правил? Який спосіб подання правил в них використовується?

*Дерево вирішальних правил – метод представлення вирішальних правил у певній ієрархії, що включає елементи двох типів - вузлів (node) і листя (leaf). Вузли включають вирішальні правила і проводять перевірку прикладів на відповідність обраного атрибуту навчальної множини.*

1. Дати означення основних понять, що відносяться до теорії дерев вирішальних правил: об’єкт, атрибут, мітка класу, вузол, лист, перевірка.

*Об’єкт – приклад, шаблон, спостереження.*

*Атрибут – ознака, незалежна змінна, властивість.*

*Мітка класу – залежна змінна, цільова змінна.*

*Вузол – внутрішній вузол дерева, вузол перевірки.*

*Лист – кінцевий вузол дерева, вузол рішення.*

*Перевірка – умова у вузлі, вирішальне правило.*

1. Навести основні класи задач, до яких можуть бути застосовані дерева вирішальних правил.
   1. *Класифікація. Віднесення об'єктів до одного із заздалегідь відомих класів. Цільова змінна повинна мати дискретні завдання.*
   2. *Регресія (чисельне передбачення). Пророцтво числового значення незалежної змінної для заданого вхідного вектора.*
   3. *Опис об'єктів. Набір правил у дереві рішень дозволяє компактно описувати об'єкти. Тому замість складних структур, що використовуються для опису об'єктів, можна зберігати дерева рішень.*
2. Яким чином відбувається побудова дерева рішень? Який метод використовується для цього?

*Основне завдання при побудові дерева рішень – послідовно і рекурсивно розбити навчальну множину на підмножини із застосуванням вирішальних правил у вузлах. Але як довго треба розбивати? Цей процес продовжують доти, доки всі вузли наприкінці гілок стануть листами.*

*Вузол стає листом у двох випадках:*

* *природним чином – коли він містить єдиний об'єкт чи об'єкт лише одного класу;*
* *після досягнення заданої умови зупинки алгоритм - наприклад, мінімально допустима кількість прикладів у вузлі або максимальна глибина дерева.*

*В основі побудови лежать «жадібні» алгоритми, що допускають локально-оптимальні рішення на кожному кроці (розбиття у вузлах), що призводять до оптимального підсумкового рішення. Тобто при виборі одного атрибуту та творі розбиття по ньому на підмножини, алгоритм не може повернутися назад та вибрати інший атрибут, навіть якщо це дасть найкраще підсумкове розбиття. Отже, на етапі побудови дерева рішень не можна точно стверджувати, що вдасться досягти оптимального розбиття.*

*Популярні алгоритми, використовуваних на навчання дерев рішень, будуються з урахуванням принципу «поділяй і владарюй».*

1. В чому полягає процес навчання з учителем?

*Вихідні дані для навчання дерева поділяються на «тренувальні» та «тестові». На основі тренувальних даних, де до кожного рядка атрибутів обов’язково наявний очікуваний результат, і відбувається навчання з учителем.*

1. Порівняйте методи, що реалізують дерева вирішальних правил: CART та C4.5.

*Алгоритм C4.5 дозволяє працювати лише з дискретною цільовою змінною. Дерева рішень, побудовані з урахуванням C4.5, виходять кваліфікуючими. Число нащадків у вузлі необмежене. Алгоритм вміє працювати із пропущеними даними.*

*Алгоритм CART вирішує завдання класифікації та регресії, оскільки дозволяє використовувати дискретну та безперервну цільові змінні. CART будує дерева, у кожному вузлі яких лише два нащадки.*

1. Поясніть принцип роботи “жадібних” алгоритмів.

*«Жадібні» алгоритми допускають локально-оптимальні рішення на кожному кроці (розбиття у вузлах), які призводять до оптимального підсумкового рішення. Тобто при виборі одного атрибуту та творі розбиття по ньому на підмножини, алгоритм не може повернутися назад та вибрати інший атрибут, навіть якщо це дасть найкраще підсумкове розбиття. Отже, на етапі побудови дерева рішень не можна точно стверджувати, що вдасться досягти оптимального розбиття.*

1. Перелічіть основні аспекти, яким приділяється увага при побудові дерев рішень.

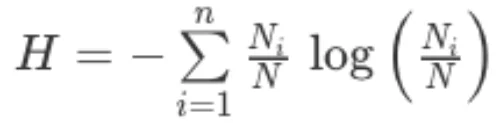
* *Вибрати атрибут для здійснення розбиття у цьому вузлі.*
* *Визначити критерій зупинення навчання.*
* *Вибрати спосіб відсікання гілок.*
* *Оцінити точність збудованого дерева.*

1. В чому полягає правило відбору ознаки для розбиття? Сформулюйте загальне правило для відбору атрибуту.

*Розбиття повинне здійснюватися за певним правилом, для якого і вибирають атрибут. Причому обраний атрибут повинен розбити безліч спостережень у вузлі так, щоб результуючі підмножини містили приклади з однаковими мітками класу або були максимально наближені до цього. Іншими словами — кількість об'єктів з інших класів у кожному з цих множин має бути якнайменше.*

1. Виконайте порівняльний аналіз критеріїв оцінки якості розбиття множини на класи.

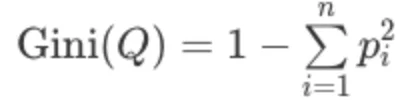
*В основі критерію ентропії (критерій номер 1) лежить інформаційна ентропія:*

**

*де n – число класів у вихідному підмножині, Ni – число прикладів i-го класу, N – загальна кількість прикладів у підмножині.*

*Ентропія сприймається як міра неоднорідності підмножини по представлених у ньому класах. І якщо класи представлені у рівних частках, а невизначеність класифікації найбільша, то ентропія теж максимальна. Логарифм від одиниці звертатиме ентропію в нуль, якщо всі приклади вузла відносяться до одного класу.*

*В основі другого методу (критерію) лежить використання індексу Джіні:*



*де Q – результуюча множина, n – число класів у ньому, pi – ймовірність i-го класу (виражена як відносна частота прикладів відповідного класу).*

*Він показує, як часто випадково вибраний приклад навчальної множини буде розпізнаний неправильно. Важлива умова — цільові значення мають братися з певного статистичного розподілу.*

*Якщо говорити простіше, то індекс Джині показує відстань між розподілами цільових значень та прогнозами моделі. Мінімальне значення показника говорить про хорошу роботу моделі.*

1. Що визначає правило зупину? Дайте порівняльну характеристику відомих критеріїв зупину побудови дерева вирішальних правил?

* *Рання зупинка. Алгоритм зупиняється після досягнення заданого значення критерію (наприклад, відсоткової частки розпізнаних прикладів). Перевага методу - скорочення тимчасових витрат за навчання. Головний недолік - рання зупинка негативно впливає на точність дерева. Через це багато фахівців радять віддавати перевагу відсіканню гілок.*
* *Обмеження глибини деревини. Алгоритм зупиняється після досягнення встановленої кількості розбиття у гілках. Цей підхід негативно позначається на точності дерева.*
* *Завдання мінімально допустимої кількості прикладів у вузлі. Встановлюється обмеження створення вузлів з числом прикладом менше заданого (наприклад, 7). У такому разі не створюватимуться тривіальні розбиття та малозначущі правила.*

1. Для чого використовується правило відсіку?

*Без обмеження «зростання» дерево рішень стане занадто великим і складним, що унеможливить подальшу інтерпретацію. А якщо робити вирішальні правила для створення вузлів, в які потраплятимуть по 2-3 приклади, вони не позбавляться практичної цінності.*

1. Що розуміють під точністю та помилкою розпізнавання для дерева вирішальних правил?

*Точність розраховується як відношення об'єктів, правильно класифікованих у процесі навчання, до кількості об'єктів набору даних, які брали участь у навчанні.*

*Помилка розраховується як відношення об'єктів, які неправильно класифіковані в процесі навчання, до загальної кількості об'єктів набору даних, які брали участь у навчанні.*

1. Що необхідно зробити для добування правил з дерева вирішальних правил?

*Для отримання правил потрібно відстежувати всі шляхи від кореневого вузла до листя дерева. Кожен шлях дає правило з безліччю умов, що є перевіркою у кожному вузлі шляху.*

*Якщо уявити складне дерево рішень у вигляді вирішальних правил (замість ієрархічної структури вузлів), воно буде простіше сприйматися та інтерпретуватися.*

1. Які вимоги до структури та значень даних висуває метод С4.5?

* *Дані би мало бути структуровані, тобто. являти собою таблицю, стовпці якої є атрибутами (ознаками), що описують предметну область або бізнес-процес, а рядки - навчальні приклади, що становлять об'єкти, що класифікуються, для яких задана мітка класу (оскільки алгоритм використовує навчання з учителем). Усі рядки повинні містити той самий набір атрибутів.*
* *Одне з атрибутів може бути визначено як цільової, тобто. атрибут класу. Для кожного навчального прикладу повинна бути задана мітка класу. Вхідні атрибути може бути як безперервними, і дискретними, а атрибут класу — лише дискретним, тобто. приймати кінцеву кількість унікальних значень.*
* *Кожен приклад навчальної множини повинен однозначно відноситися до відповідного класу. Імовірнісні оцінки ступеня належності прикладів до класу не використовуються (така постановка відноситься до нечітких дерев рішень). Число класів у навчальній множині має бути набагато менше кількості навчальних прикладів.*

1. Проаналізуйте алгоритм побудови дерева вирішальних правил за допомогою методу С4.5.

*На першому етапі навчання формується «порожнє» дерево, яке складається тільки з кореневого вузла, що містить все навчальну множину. Потрібно розбити кореневий вузол на підмножини, з яких будуть сформовані вузли-нащадки. Для цього вибирається один з атрибутів і формуються правила, які розбивають навчальну множину на підмножини, число яких дорівнює кількості p унікальних значень атрибута.*

*В результаті розбиття виходять p (за кількістю значень атрибуту) підмножин і, відповідно, формуються p нащадків кореневого вузла, кожному з яких поставлено у відповідність своє підмножина. Потім ця процедура рекурсивно застосовується до всіх підмножин доти, доки не буде виконано умову зупинки навчання.*

1. Яким чином визначається критерій вибору атрибуту в методі С4.5?

*Для вибору найкращого атрибута розгалуження можна використовувати наступний критерій:*

Gain(A)=Info(S)−InfoA(S), *—* *критерій приросту інформації (від англ. Gain - приріст, збільшення). Потім значення критерію обчислюється всім потенційних атрибутів розбиття, і вибирається той атрибут, який максимізує його.*

1. В якому випадку в процесі роботи методу С4.5 вузол помічається як лист? Що обирається в якості розв’язку листа?

*Якщо в процесі побудови дерева буде сформований «порожній» вузол, куди не потрапило жодного прикладу, він перетворюється на лист, який асоціюється з класом, що найчастіше зустрічається у безпосереднього предку вузла. В якості розв’язку обирається більший параметр.*

1. Коли ентропія досягає свого максимуму (мінімуму) при використанні методу С4.5?

*Максимум – в корені дерева, а мінімум – на листях.*

1. В яких випадках необхідно обрати поріг для порівняння значень атрибуту?

*У разі безперервних атрибутів вибирається поріг, з яким порівнюватимуть усі значення.*

1. В чому полягає класифікація нових об’єктів? Звідки починається обхід дерева?

*Якщо на якомусь вузлі дерева при виконанні перевірки з'ясовується, що значення відповідного атрибута прикладу, що класифікується, пропущено, то алгоритм досліджує всі можливі шляхи вниз по дереву і визначає з якою ймовірністю приклад відноситься до різних класів. У цьому випадку «класифікація» — це скоріше розподіл класів. Як тільки розподіл класів встановлено, то клас, що має найбільшу ймовірність появи, вибирається як відповідь дерева рішень.*

1. Порівняйте покращений критерій розбиття з класичним.

*Новий критерій дозволяє оцінити частку інформації, отриманої під час розбиття, яка є «корисною», тобто сприяє покращенню класифікації. Застосування цього показника, зазвичай, призводить до вибору більш вдалого атрибуту, ніж використання звичайного критерію приросту інформації.*

*Сенс цієї модифікації є досить простим. N(Si)N(S) — відношення числа прикладів у підмножині, отриманому в результаті розбиття, до прикладів у батьківській множині SS. Якщо в результаті розбиття виходить велика кількість підмножин з невеликою кількістю прикладів, що характерно для перенавчання, показник Split-Info зростає.*

*Оскільки він стоїть у знаменнику виразу, Gain-Ratio є звичайний приріст інформації, «оштрафований» за допомогою Split-Info. Завдяки цьому атрибут, для якого Split-Info росте, має менше шансів бути обраним для розбиття, ніж при використанні звичайного Gain-критерію*.

1. Яке евристичне правило використовується для зменшення ймовірності створення вузлів та листя, які містять незначну кількість об’єктів?

*При розбитті множини S принаймні два підмножини повинні мати не менше заданої мінімальної кількості прикладів u (u>1), зазвичай рівного 2.*

1. Проаналізуйте процедуру роботи з пропущеними даними.

*Підхід можна сформулювати так: передбачається, що пропущені значення атрибуту розподілені пропорційно частоті появи існуючих значень.*

1. Яким чином відбувається класифікація нових об’єктів у випадку відсутності значення певного атрибуту об’єкту, що класифікується?

*Якщо на якомусь вузлі дерева при виконанні перевірки з'ясовується, що значення відповідного атрибута прикладу, що класифікується, пропущено, то алгоритм досліджує всі можливі шляхи вниз по дереву і визначає з якою ймовірністю приклад відноситься до різних класів.*

1. Які переваги використання дерев вирішальних правил?

* *Формують чіткі та зрозумілі правила класифікації. Наприклад, «якщо вік < 40 і немає майна для застави, то відмовити у кредиті». Тобто дерева рішень добре та швидко інтерпретуються.*
* *Чи здатні генерувати правила в областях, де фахівцю важко формалізувати свої знання.*
* *Легко візуалізуються, тобто можуть «інтерпретуватися» не лише як модель загалом, а й як прогноз окремого тестового суб'єкта (шлях у дереві).*
* *Швидко навчаються та прогнозують.*
* *Не потрібно багато параметрів моделі.*
* *Підтримують як числові, і категоріальні ознаки.*

1. Який напрямок побудови дерева вирішальних правил використовується при використанні методу ID3?

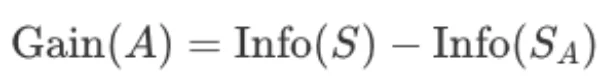
*Напрямок побудови: від кореня до листя.*

1. Наведіть послідовність побудови дерева вирішальних правил за допомогою методу ID3.
   1. *Взяти всі невикористані ознаки та порахувати їхню ентропію щодо тестових зразків*
   2. *Вибрати ознаку, для якої ентропія мінімальна (а інформаційна вигода відповідно максимальна)*
   3. *Зробити вузол дерева, що містить цю ознаку*
2. В чому полягає вибір властивості на основі теорії інформації?

*Ентропія сприймається як міра неоднорідності підмножини по представлених у ньому класах. І якщо класи представлені у рівних частках, а невизначеність класифікації найбільша, то ентропія теж максимальна. Логарифм від одиниці звертатиме ентропію в нуль, якщо всі приклади вузла відносяться до одного класу.*

*Якщо обраний атрибут розбиття Aj забезпечує максимальне зниження ентропії результуючого підмножини щодо батьківського, його можна вважати найкращим.*

*Але насправді про ентропію говорять рідко. Фахівці приділяють увагу зворотній величині – інформації. У такому разі найкращим атрибутом буде той, що забезпечить максимальний приріст інформації результуючого вузла щодо вихідного:*

**

*де Info(S) - інформація, пов'язана з підмножиною S до розбиття, Info(Sa) - інформація, пов'язана з підмножиною, отриманою при розбитті атрибута A.*

*Завдання вибору атрибута в такій ситуації полягає у максимізації величини Gain(A), яку називають приростом інформації. Тому теоретико-інформаційний підхід також відомий за назвою «критерій приросту інформації.*

1. Проаналізуйте основні функції пакету Matlab для роботи з деревами вирішальних правил: внутрішня структура, параметри, основні змінні, методи, їх призначення та використання.

**6.2. Висновки**

В результаті виконання лабораторної роботи ми ознайомилися з деревами рішень та з бібліотекою sklearn для мови програмування python, завдяки якої були створені дерева рішень для чотирьох наборів даних.

Для першого набору даних (Iris) побудоване дерево не вдалося спростити методом “cost complexity prunning”, адже початкова функція побудувала уже максимально спростоване дерево та знайдений вартісний параметр складності ccp\_alpha дорівнював 0.

Для другого набору даних (Digits) було побудоване дуже велике дерево рішень, адже це досить великий набір даних з великим набором атрибутів. Але його вдалося спростити і навіть підвищити точність приблизно на 0.01 при ccp\_alpha = 0.0007.

Для третього набору даних (Wine) побудоване дерево було зовсім трохи спростовано, адже найбільш ефективним значенням ccp\_alpha = 0.01344 виявилося 2 серед усіх можливих. Тому це спрощення навіть не вплинуло на точність дерева.

Для четвертого набору даних (Breast cancer) створене дерево вдалося дуже добре спростити та підвищити точність дерева на 0.015 при тому що глибина дерева зменшилася більше ніж у два рази. Знайдено ефективне значення ccp\_alpha = 0.0085.