Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
магістр
(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ ВІДБОРУ ОЗНАК
RESEARCH AND SOFTWARE IMPLEMENTATION OF FEATURE SELECTION

Виконав(ла): студент(ка) 2 курсу, групи КНТ-214м
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

КИРИК М.М.
(ПРІЗВИЩЕ та ініціали)
Керівник ІЛЬЯШЕНКО М.Б.
(ПРІЗВИЩЕ та ініціали)
Рецензент КОЦУР М.І.
(ПРІЗВИЩЕ та ініціали)

2025

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Системи штучного інтелекту
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____Сергій СУББОТІН
"____" _____2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

КИРИКА Максима Миколайовича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів відбору ознак. Research and Software Implementation of Feature Selection

керівник проєкту (роботи) к.т.н., доцент, ІЛЬЯШЕНКО Матвій Борисович ,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)
затверджені наказом закладу вищої освіти від " 30 " вересня 2025 року № 447
2. Строк подання студентом проєкту (роботи) 02 грудня 2025 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ІЛЬЯШЕНКО М.Б., доцент		
Нормоконтроль	ДЕЙНЕГА Л.Ю., ст. викладач		

7. Дата видачі завдання “ 30 ” вересня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки.		
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження	8 тиждень	Розділ 4
	програмного забезпечення.		
7	Оформлення пояснювальної записки та документів	9-10 тижні	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка) _____ Максим КИРИК
(підпис) (Імя ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ Матвій ІЛЬЯШЕНКО
(підпис) (Імя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра: 91 с., 4 табл., 25 рис., 3 дод., 19 джерел.

PYCHARM, PYTHON, ВІДБІР ОЗНАК, ГЕНЕТИЧНИЙ АЛГОРИТМ, МОВА ПРОГРАМУВАННЯ, ОЗНАКА, ОПТИМІЗАЦІЯ, ЦІЛЬОВА ФУНКЦІЯ.

Об’єкт дослідження – процеси обчислень, пов'язані з відбором інформативних ознак.

Предмет дослідження – методи відбору інформативних ознак.

Мета роботи – дослідження та реалізація методів відбору ознак.

Матеріали, методи та технічні засоби: мова програмування Python, середовище розробки PyCharm.

Результати. Розроблено програмне забезпечення для обчислень, що пов'язані з відбором інформативних ознак, використовуючи мову програмування Python та середовище розробки PyCharm.

Практична цінність роботи полягає у розробці програмного забезпечення для обчислень, пов'язаних з відбором інформативних ознак.

Висновки. Виконано проєктування програмного забезпечення для обчислень, пов'язаних з відбором інформативних ознак. Розроблено програмне забезпечення для обчислень, що пов'язані з відбором інформативних ознак. Здійснено тестування розробленого програмного забезпечення для відбору ознак.

Галузь використання – розпізнавання образів, прогнозування, аналіз даних.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 91 pages, 4 tables, 25 figures, 3 appendixes, 19 sources.

PYCHARM, PYTHON, FEATURE SELECTION, GENETIC ALGORITHM, PROGRAMMING LANGUAGE, FEATURE, OPTIMIZATION, OBJECTIVE FUNCTION.

The object of research is the process of computations related to the selection of informative features.

The subject of the research is methods of selecting informative features.

The purpose of this work is research and implementation of methods of selecting informative features.

Materials, methods and technical tools: Python programming language, PyCharm development environment.

Results. The software tools for calculations related to the selection of informative features using the Python programming language and the PyCharm development environment have been developed.

The practical value of the work lies in the development of software for calculations related to the selection of informative features.

Conclusions. The software for calculations related to the selection of informative features has been designed. The software for calculations related to the selection of informative features has been developed. The developed software for feature selection has been tested.

Scope of use – pattern recognition, forecasting, data analysis.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ВО – відбір ознак;

ГА – генетичний алгоритм;

ПЗ – програмне забезпечення;

ЦФ – цільова функція.

ВСТУП

Відбір ознак є одним із ключових етапів у процесі інтелектуальних обчислень, що безпосередньо впливає на точність, узагальнювальну здатність та обчислювальну ефективність моделей. Наявність у вихідних даних надмірних, корельованих або малозначущих характеристик зумовлює підвищений ризик перенавчання, ускладнює інтерпретацію результатів і збільшує витрати обчислювальних ресурсів. У зв’язку з цим актуальність проблематики відбору ознак зумовлюється необхідністю формування компактних та інформативних підмножин характеристик, які забезпечують побудову більш стійких та продуктивних моделей обробки інформації [1]-[3].

Розробка програмного забезпечення для реалізації методів відбору ознак набуває особливої ваги, оскільки практичне застосування даних підходів у реальних системах вимагає автоматизації процесів, уніфікації алгоритмічних рішень та інтеграції з існуючими інструментами аналізу. Використання спеціалізованих програмних засобів дозволяє підвищити відтворюваність експериментів, стандартизувати процедури обробки даних і створити масштабовані платформи для вирішення задач у різних сферах — від економіки та медицини до технічних наук. Таким чином, поєднання методологічних підходів до оптимізації простору ознак із програмною реалізацією розглядається як невід’ємна складова розвитку сучасних інтелектуальних систем, що визначає ефективність їх подальшого використання у практичних застосуваннях [4]-[6].

Програмне забезпечення, що реалізує методи відбору ознак, попри свою практичну цінність, має низку обмежень, які впливають на якість та ефективність його використання. Складність деяких методів призводить до значних обчислювальних витрат, що стає критичним при роботі з великими масивами даних. Крім того, реалізовані програмні рішення не завжди забезпечують достатню гнучкість, оскільки часто орієнтовані на використання певних підходів і можуть мати обмежені можливості налаштування для

специфічних завдань. Ще однією суттєвою проблемою є ризик втрати важливої інформації, коли алгоритми відбору зменшують простір ознак надмірно агресивно, і, як наслідок, модель втрачає здатність до адекватного відображення складних залежностей. В окремих випадках труднощі викликає також інтерпретація результатів, оскільки автоматизовані системи не завжди надають зрозуміле пояснення того, чому були відкинуті певні характеристики. Обмеженість інтеграційних можливостей із зовнішніми аналітичними програмними платформами та недостатня оптимізація під конкретні архітектури обчислювальних систем також ускладнюють практичне застосування таких рішень. Таким чином, хоча програмні засоби для відбору ознак є важливим інструментом у сучасному аналізі даних, їх використання супроводжується низкою недоліків, які потребують подальшого дослідження та вдосконалення [1]-[6].

Тому актуальним є дослідження та реалізація методів відбору ознак. У дипломній кваліфікаційній роботі магістра розв’язується актуальне завдання дослідження та реалізації методів відбору ознак.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв’язати такі задачі:

– виконати методів та засобів відбору ознак;

– розробити метод відбору інформативних ознак;

– здійснити проєктування програмного забезпечення, що реалізує методи відбору ознак;

– створити програмне забезпечення, що реалізує методи відбору ознак;

– виконати тестування розробленого програмного забезпечення, що реалізує методи відбору ознак.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз методів та засобів відбору ознак

Відбір ознак є підходом до скорочення розмірності, що передбачає вибір обмеженої підмножини релевантних ознак з первинного набору даних шляхом усунення шумових, надлишкових або малозначущих елементів. Виконання такого відбору здатне підвищити ефективність навчання моделей, забезпечити більшу точність прогнозування, зменшити обчислювальні витрати та полегшити інтерпретацію результатів [1].

Проблема відбору ознак стає критичною у випадках, коли кількість ознак порівнянна або перевищує обсяг вибірки, що призводить до перенавчання моделі та зниження її продуктивності на тестових даних. Додатково, моделювання з великими наборами ознак вимагає значних обчислювальних ресурсів і часу [1].

Відбір ознак визначається через оцінку релевантності та надмірності у відношенні до цільової змінної. Підмножина ознак формується на основі цих характеристик, що дозволяє виділяти категорії ознак за їхньою значимістю та корисністю. Ознаки, що не впливають на точність прогнозу, вважаються нерелевантними, тоді як слаборелевантні елементи можуть бути включені або виключені в залежності від специфіки задачі. Надлишкові ознаки здатні дублювати інформацію інших змінних без порушення розподілу цільової змінної, і їхня перевірка стає необхідною при багатовимірному аналізі підмножини [1].

Взаємозв’язок між ознаками визначається статистичними методами, що дозволяє вимірювати їхню релевантність незалежно від окремих значень. Спотворення результатів може виникати не через непотрібну інформацію, а через відсутність статистичного зв’язку з іншими ознаками. Водночас одна ознака може виявитися нерелевантною самостійно, але суттєвою у комбінації з іншими елементами [1], [2].

Сильно релевантні вхідні ознаки завжди залишаються критично необхідними для формування оптимальної підмножини, оскільки їхнє видалення змінює умовний розподіл цільової змінної. Основна мета відбору ознак полягає у максимізації релевантності при одночасному мінімізації надлишковості [1], [2].

Використання методів відбору ознак у підготовці даних дозволяє ефективно скорочувати обсяг інформації та забезпечує пошук точних моделей. Оскільки повний перебір оптимальної підмножини часто неможливий, застосовуються різні стратегії пошуку. Вибір ознак застосовується в задачах класифікації, регресії та кластеризації для підвищення продуктивності алгоритмів [1], [2].

Відбір ознак може розглядатися як важливий етап інженерії ознак у процесах машинного навчання, який зосереджується на виділенні найбільш інформативних та ненадлишкових змінних з наявного набору даних. Він відрізняється від вилучення ознак, оскільки не створює нових змінних на основі знань предметної області, а фокусується на оцінці вже наявних ознак та визначенні їхньої значущості для моделі. Основна мета цього процесу полягає у відборі таких підмножин ознак, які забезпечують максимальну релевантність, мінімальну надмірність і покращують продуктивність навчання моделей [1], [2].

Використання методів відбору ознак дозволяє скоротити обсяг даних для обробки, зменшити складність моделей, прискорити навчання та знизити ризик перенавчання. Надмірність даних у сучасних великих наборах змушує застосовувати розумні алгоритми для виявлення критично важливої інформації, що забезпечує ефективне формування моделей у задачах класифікації, регресії або кластеризації. Попередня обробка даних стає необхідною не лише для зменшення розміру набору, але й для оптимального узгодження даних із обраними методами аналізу [1], [2].

Якість представлення ознак визначає ефективність моделей машинного навчання. Незалежно від підходу до створення векторів ознак, важливо

забезпечити, щоб вибрані ознаки могли відокремлювати причинно-наслідкові фактори, були простими для моделювання та добре інтегрувалися зі стратегіями регуляризації. Надійне представлення ознак включає такі змінні, які відображають основні фактори, що формують спостережувані дані, забезпечуючи розділення різних причин. Простота моделювання дозволяє алгоритмам швидше знаходити закономірності та будувати стабільні прогностичні моделі, а сумісність із регуляризаційними методами сприяє кращому узагальненню та зниженню ризику перенавчання [1], [2].

Структурування та організація відобраних ознак є ключовим фактором для підвищення якості даних та їхнього ефективного використання у процесах автоматизованого прийняття рішень. Надійні представлення дозволяють поєднувати високу точність моделювання з можливістю інтерпретації результатів і подальшого використання в аналітичних процесах, що робить їх невід'ємною складовою сучасних конвеєрів машинного навчання [1], [2].

Відбір ознак може розглядатися через призму наявності або відсутності інформації про цільові мітки, що дозволяє виділити три основні підходи: повністю керовані алгоритми, напівкеровані методи та підходи без використання міток. Керовані методи опираються на наявні мітки для визначення найбільш інформативних і дискримінативних ознак, які здатні відокремлювати об'єкти різних класів. Вони орієнтуються на досягнення цілей моделей із мітками, наприклад, у задачах класифікації або регресії, використовуючи доступні марковані дані для оцінки релевантності кожної ознаки [1], [2].

Методи відбору ознак без використання міток працюють незалежно від цільової змінної та зазвичай базуються на статистичних або кореляційних критеріях для видалення надлишкових або нерелевантних ознак. Напівкеровані підходи до відбору інформативних ознак застосовуються, коли доступна лише частина маркованих даних, і використовують як позначені, так і непозначені приклади для побудови матриці подібності, яка допомагає відібрати найбільш узгоджені ознаки. У порівнянні з керованими методами,

безміткові підходи вважаються складнішими через відсутність чіткої орієнтації на дискримінативні ознаки [1], [2].

У межах керованих методів відбір ознак поділяється на кілька категорій, які визначають взаємодію з моделлю навчання: методи фільтрації, обгортки, гібридні та вбудовані. Метод фільтрації визначає важливість ознак на основі статистичних показників, таких як інформаційний приріст, тест хі-квадрат, коефіцієнт кореляції або дисперсія. Ознаки ранжуються за цими метриками, що дозволяє видаляти нерелевантні або надлишкові змінні та покращувати точність моделі без суттєвого навантаження на обчислювальні ресурси [1], [2].

Інші показники, які враховуються при відборі ознак, включають відсутні значення, які оцінюються як частка від загальної кількості спостережень і визначають, які ознаки варто виключити. Важливість кожної ознаки може оцінюватися за зміною ентропії, здатністю відокремлювати класи або за кореляційними властивостями щодо цільової змінної, з урахуванням мінімальної взаємної кореляції між ознаками. Методи на кшталт низької дисперсії, середньої абсолютної різниці дозволяють визначати релевантність ознак на основі їхньої варіативності, тоді як метод перестановки оцінює важливість ознаки через вплив її випадкового перемішування на продуктивність моделі. Цей підхід не залежить від типу моделі і може застосовуватися для складних або нелінійних оцінок, надаючи гнучкий інструмент для виділення ключових змінних у наборі даних [1], [2].

Підхід фільтраційного методу Relief вирізняється високою чутливістю до взаємодії між ознаками. Його спочатку застосовували для задач бінарної класифікації з числовими або дискретними ознаками. Кожна ознака отримує власну оцінку, що дозволяє впорядкувати їх за значущістю і вибирати найбільш інформативні для подальшого використання у моделюванні. Оцінювання здійснюється через аналіз відмінностей у значеннях ознак між парами найближчих сусідів: якщо зміни спостерігаються в парах з однаковим класом, значимість ознаки зменшується, а якщо у парах з різними класами —

збільшується. Цей базовий алгоритм став основою для створення сімейства алгоритмів відбору ознак, які були адаптовані для стабільнішого функціонування в умовах високого рівня шуму, для роботи з багатокласовими задачами та регресійними завданнями, а також для коректної роботи при наявності неповних даних [1], [2].

Обгорткові методи підходять до задачі відбору ознак як до процесу пошуку, у якому різні комбінації підмножин ознак створюються, оцінюються та порівнюються між собою. Оцінка кожної комбінації здійснюється за допомогою прогнозної моделі, що дозволяє визначати їхню ефективність у контексті конкретного завдання. Точність і вибір оптимальної підмножини ознак залежать від обраного класифікатора, оскільки продуктивність підмножини визначається результатами його роботи. Методи обгортки вимагають значних обчислювальних ресурсів через багаторазове навчання моделей і проведення перехресної перевірки, але забезпечують вищу точність порівняно з методами фільтрації. До них відносяться різноманітні підходи, включно з рекурсивним виключенням ознак, алгоритмами послідовного вибору та генетичними алгоритмами [2], [3].

Алгоритм Boruta реалізує обгортковий підхід, використовуючи класифікатор XGBoost для оцінки важливості ознак. Спочатку створюються тіньові ознаки шляхом дублювання існуючих та перемішування їхніх значень, після чого класифікатор навчається на даних кілька разів для визначення ваги кожної ознаки. Контрольний показник формується на основі середньої значущості тіньових ознак, і лише справжні ознаки, значимість яких перевищує цей показник, залишаються для подальшого аналізу [1]-[3].

Ітеративний підхід прямого вибору ознак передбачає поетапне додавання ознак із найвищою значущістю щодо цільової змінної, при цьому кожна наступна ознака підбирається так, щоб підвищувати продуктивність у поєднанні з уже відібраними. Зворотне виключення ознак діє навпаки: всі ознаки використовуються на початку, після чого на кожному кроці виключається найменш значуща, і модель перебудовується на залишку до

завершення процесу. Вичерпний підхід оцінює всі можливі комбінації ознак, обираючи оптимальну підмножину за результатами повного перебору. Рекурсивне виключення ознак застосовує оцінки важливості, отримані від моделі, для поетапного видалення найменш значущих ознак, повторюючи цей процес рекурсивно, поки не досягається необхідна кількість ознак, забезпечуючи таким чином поступове скорочення набору при збереженні продуктивності моделі [2], [3].

Гібридний підхід до відбору ознак реалізується шляхом комбінування різних методів з метою підвищення ефективності процесу. Спершу визначаються методи ранжування, які дозволяють створити список ознак за їхньою важливістю. Потім використовується обмежена кількість найвищих за рейтингом ознак для застосування методів обгортки, що дозволяє значно скоротити простір ознак і знизити обчислювальні витрати при виконанні обгорткових процедур, одночасно зберігаючи високу продуктивність моделі [2], [3].

Вбудовані методи інтегрують відбір ознак безпосередньо в процес навчання моделі, що забезпечує більш оптимальні результати порівняно з класичними підходами фільтрації чи обгортки. Алгоритми, такі як дерева рішень, виконують розподіл набору даних на підмножини на кожному кроці росту дерева, автоматично виділяючи найбільш релевантні ознаки. Регуляризаційні підходи дозволяють зменшувати вагу деяких коефіцієнтів до нуля, що дає змогу видаляти нерелевантні ознаки без шкоди для моделі. У випадку ансамблевих методів, таких як випадкові ліси, важливість ознак визначається через покращення чистоти вузлів і зменшення домішок, причому найзначущі ознаки розташовуються ближче до кореня дерев, що дозволяє формувати підмножину ключових ознак шляхом обрізання менш інформативних гілок [2], [3].

Методи відбору ознак без учителя застосовуються у випадках, коли марковані дані відсутні або їх мало, що робить їх популярними для роботи з високовимірними наборами даних. Такі підходи зазвичай оцінюють

важливість ознак через здатність зберігати структуру даних, при цьому часто не враховується надмірність між ознаками, а також залишається відкритим питання визначення оптимальної кількості ознак [2], [3].

При створенні автоматизованих алгоритмів вибору підмножини ознак для немаркованих даних стикаються з проблемами поєднання пошуку кращої кількості кластерів з відбором ознак та нормалізації критеріїв вибору з урахуванням розмірності даних. Методи відбору ознак без учителя можуть реалізовуватися за принципом фільтрації, обгортки або гібридного підходу, залежно від способу взаємодії з моделлю навчання [3], [4].

Фільтраційні методи без учителя поділяються на одновимірні та багатовимірні. Одновимірні підходи оцінюють кожну ознаку окремо, формуючи рейтинг за релевантністю, що дозволяє відсіяти явно нерелевантні ознаки, проте надлишкові залежності залишаються непоміченими. Багатовимірні методи враховують взаємозв’язки між ознаками, що дає змогу ефективніше відбирати підмножини, які одночасно мінімізують надлишковість і зберігають корисну інформацію для подальшого навчання моделі [3], [4].

Одновимірні підходи можуть базуватися на інформаційних критеріях або на спектральному аналізі даних. Інформаційні методи оцінюють ступінь розсіювання даних через ентропію, взаємну інформацію та інші міри, щоб визначити структури та кластери в даних. Спектральні методи моделюють локальні чи глобальні структури даних через лапласіанські або нормалізовані лапласіанські матриці, побудовані на основі матриці подібності об’єктів [3], [4].

Методи на основі ентропії, як-от послідовний зворотний вибір, визначають релевантність кожної ознаки через вплив на загальну ентропію, розраховану за матрицею подібності між об’єктами. При низькій ентропії дані демонструють чіткі кластерні структури, а високі значення вказують на їх відсутність. Внесок кожної ознаки оцінюється шляхом поетапного

виключення, що дозволяє отримати ранжування від найрелевантнішої до найменш релевантної [3], [4].

Інший метод базується на аналізі сингулярних значень матриці даних, щоб визначити ознаки, що найкраще відображають структуру даних. При низькій ентропії формується чітко виражена кластерна структура, а при високій – спектр розподілений більш рівномірно, що ускладнює виділення кластерів. Внесок кожної ознаки у зміну ентропії визначається шляхом порівняння її присутності або виключення. Інший підхід оцінює здатність ознаки стиснути інформацію в наборі даних через власні значення коваріаційної матриці. Значення цієї ентропії коливається від нуля до одного, де 1 означає максимальне стиснення інформації, а 0 – мінімальне, що дозволяє відокремити найбільш інформативні ознаки для подальшого аналізу [3], [4].

Методи відбору ознак на основі спектральної подібності оцінюють важливість ознак через їхню здатність відображати структуру даних у графі, побудованому на основі подібності об’єктів. Лапласова оцінка визначає релевантність ознаки через дисперсію та здатність підтримувати локальні зв’язки між близькими об’єктами, при цьому ознаки, що приймають подібні значення для сусідніх елементів і відмінні для віддалених, отримують високі ваги. Цей підхід забезпечує збереження локальної структури графа, яка відображає ймовірність належності об’єктів до одного кластеру [4], [5]. Інший метод оцінює узгодженість ознаки з власними векторами матриці Лапласа графа подібності. Процес включає побудову матриці подібності, обчислення нетривіальних власних векторів та ранжування ознак за ступенем відповідності цим векторним представленням. На практиці такий підхід розширює ідею лапласової оцінки, дозволяючи точніше визначати релевантні ознаки. Неконтрольований спектральний вибір ознак аналізує вплив кожної ознаки на спектр нормалізованої лапласіанської матриці, оцінюючи зміну перших нетривіальних власних значень і формуючи рейтинг ознак на основі величини спектральних прогалин [4], [5].

Багатовимірні методи фільтрації застосовують різні принципи для відбору підмножин ознак. Статистично-інформаційні підходи базуються на дисперсії, кореляції, ентропії або взаємній інформації для оцінки важливості ознак. Методи на основі природної еволюції використовують стохастичні пошукові стратегії, наприклад ройовий інтелект, щоб знайти оптимальні підмножини. Спектральні та розріджені методи поєднують спектральний аналіз із розрідженими моделями, інколи інтегруючи відбір ознак безпосередньо в процес навчання моделі, що робить їх подібними до вбудованих методів [4], [5].

Серед багатовимірних підходів також виділяються інформаційні методи, які оцінюють релевантність через ентропію подібності між об’єктами, а статистичні методи мінімальної залежності прагнуть зменшити надлишковість, підбираючи незалежні ознаки так, щоб лінійна залежність між ними була мінімальною [4], [5].

Біологічні методи відбору інформативних ознак, наприклад оптимізація колонії мурах, моделюють простір ознак як граф, де вузли відповідають ознакам, а ребра відображають їхню подібність. Ознаки, що демонструють високу схожість, вважаються надлишковими. Вибір підмножин відбувається через ітеративне переміщення агентів по графу, оновлення значень феромонів і пріоритет на ознаки з низькою надмірністю. Такі підходи забезпечують відбір ознак із високою інформативністю та мінімальною повторюваністю, що може застосовуватися як у контексті мікрочипів, так і для загального відбору релевантних ознак у даних [5], [6].

Вибір ознак у багатокластерних даних реалізується через поєднання спектрального аналізу, розрідженого навчання та відбору на основі отриманих ваг. Спочатку структура кластерів у наборі даних визначається шляхом спектрального аналізу, що дозволяє виділити ключові напрямки варіацій у даних. Далі релевантність ознак оцінюється через регресійну модель з L1-регуляризацією, що забезпечує визначення значущих ознак на основі величини коефіцієнтів. Завершальним кроком є вибір ознак із найвищими

абсолютними коефіцієнтами, що формує підмножину найважливіших елементів набору даних [5], [6].

Обгорткові методи відбору інформативних ознак без учителя базуються на оцінці підмножин ознак через результати певного алгоритму кластеризації. Основною метою таких підходів є знаходження наборів ознак, які покращують якість кластеризації, проте через численні обчислювальні цикли ці методи зазвичай є ресурсомісткими і тісно прив’язані до конкретного алгоритму кластеризації. Вибір ознак у цьому підході організовується за різними стратегіями пошуку: послідовною, біоінспірованою та ітеративною. Послідовний підхід реалізує додавання або видалення ознак по черзі, що забезпечує простоту і швидкість виконання. Біоінспіровані методи вводять випадковість у процес пошуку, прагнучи уникнути локальних оптимумів. Ітеративні стратегії розглядають відбір ознак як задачу оцінювання, що дозволяє уникнути прямого комбінаторного перебору всіх можливих підмножин [5], [6].

У послідовних підходах оцінка підмножин ознак відбувається через критерії максимальної правдоподібності або роздільності розсіювання. Підмножини кандидатів піддаються кластеризації за допомогою алгоритмів типу K-середніх, а отримані кластери оцінюються за обраними критеріями. Вибір ознак завершується, коли зміна значення критеріїв стає незначною. Подібний підхід реалізується й через спрощений силуетний метод, де підмножини оцінюються на основі якості кластерів, визначеної спрощеним показником силуету, і відбираються ті ознаки, що забезпечують найвищі значення цього показника [5], [6].

Ітеративні методи оцінюють важливість ознак шляхом присвоєння кожній з них ваг, що відображають релевантність у межах кластерів. Значущість визначається через модифіковані алгоритми EM, які моделюють компоненти даних і повертають параметри функцій щільності разом із набором ваг. Ознаки з найвищими значеннями вважаються найбільш інформативними для розрізнення кластерів [6], [7].

Гібридні методи відбору ознак поєднують властивості підходів фільтра та обгортки з метою досягнення балансу між обчислювальною ефективністю та якістю результатів при використанні відібраних ознак. На початковому етапі фільтрації ознаки оцінюються за внутрішніми характеристиками даних, формуючи ранжований або попередньо відфільтрований набір, тоді як на етапі обгортки підмножини цих ознак піддаються оцінці за допомогою спеціалізованого алгоритму кластеризації для визначення найкращого набору. Такі методи можуть відрізнятися тим, чи використовують вони ранжування ознак, чи працюють без нього. У науковій літературі існують приклади гібридних методів для конкретних областей обробки даних, розроблених з урахуванням специфіки задач [6], [7].

Підхід на основі вимірювання ентропії поєднує фільтраційний етап із критерієм роздільності внутрішнього розсіювання. Кожна ознака по черзі видаляється, і оцінюється внесок її виключення в зміну ентропії набору даних, що формує ранжований список ознак. Потім на етапі обгортки застосовується прямий пошук у поєднанні з алгоритмом k-середніх для створення кластерів і визначення підмножини ознак, яка забезпечує найвищу якість кластеризації за заданим критерієм [6], [7].

Визначено, що відбір ознак є підходом до скорочення розмірності, що передбачає вибір обмеженої підмножини релевантних ознак з первинного набору даних шляхом усунення шумових, надлишкових або малозначущих елементів. Виконання такого відбору здатне підвищити ефективність навчання моделей, забезпечити більшу точність прогнозування, зменшити обчислювальні витрати та полегшити інтерпретацію результатів. Відзначено, що надмірність даних у сучасних великих наборах змушує застосовувати розумні алгоритми для виявлення критично важливої інформації, що забезпечує ефективне формування моделей у задачах класифікації, регресії або кластеризації. Використання методів відбору ознак дозволяє скоротити обсяг даних для обробки, зменшити складність моделей, прискорити навчання та

знизити ризик перенавчання. Проаналізовано методи відбору інформативних ознак.

1.2 Програмне забезпечення для аналізу даних та відбору ознак

Проаналізуємо програмне забезпечення для аналізу даних та відбору інформативних ознак [8]-[15].

Програмне забезпечення Rapid Insight Veera (рис. 1.1) призначене для перетворення неструктурованих даних у зрозумілу та корисну інформацію, що може безпосередньо використовуватися у процесі прийняття рішень. Завдяки інтуїтивному візуальному середовищу воно усуває потребу у складному програмуванні та дозволяє швидко організовувати повторювані процедури аналізу. Робота з інформацією здійснюється у наочній формі, що значно спрощує процес підготовки, обробки та відбору інформативних ознак для подальшого моделювання [8], [13].

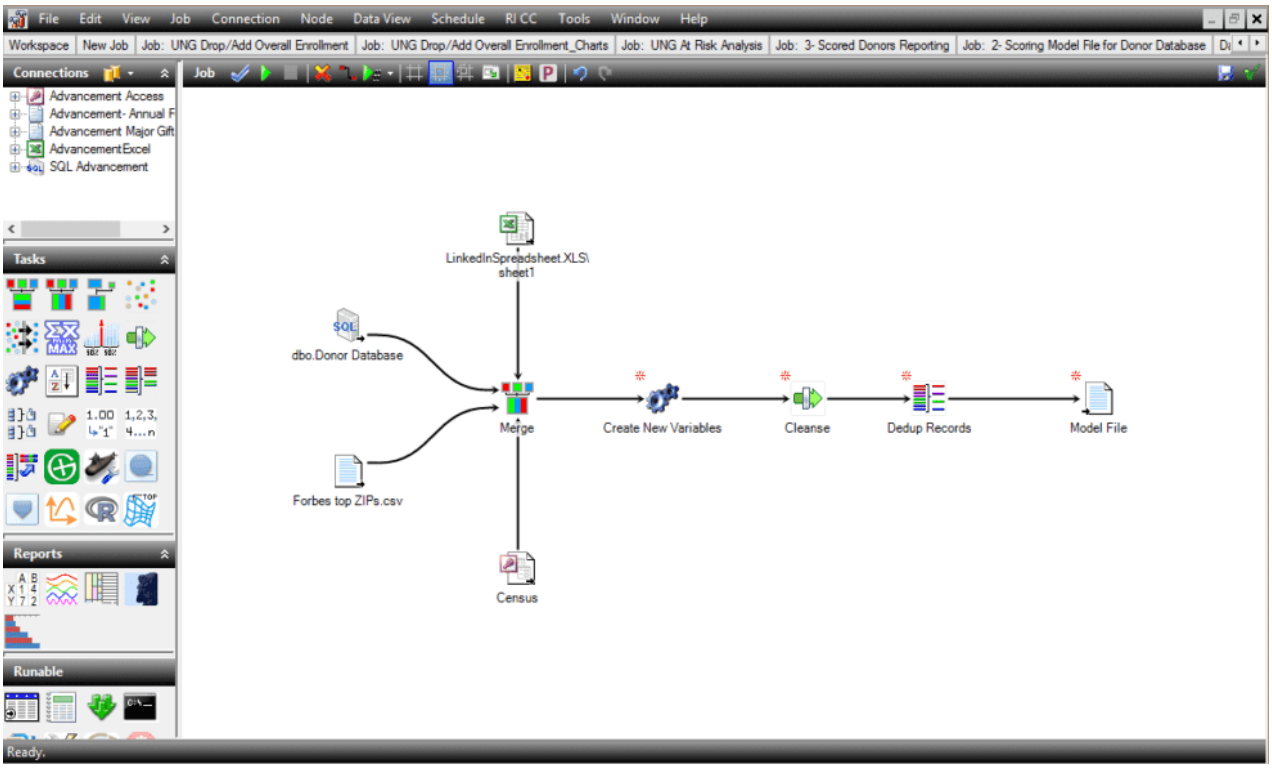


Рисунок 1.1 – Програмне забезпечення Rapid Insight Veera [13]

Програмне забезпечення Rapid Insight Veera підтримує інтеграцію даних із найрізноманітніших джерел, зокрема таблиць, баз даних, текстових масивів та хмарних сховищ. Усі дані можуть бути легко об’єднані, трансформовані й агреговані, що створює основу для подальших прогнозних досліджень та статистичного аналізу. Це дозволяє організувати повний цикл підготовки інформації — від завантаження вхідних даних до формування готових наборів для моделювання [8], [13].

Використання Veera забезпечує ефективність як для регулярної підготовки звітів, так і для виконання складних досліджень. Завдяки можливості створення візуального робочого процесу стає можливим не лише організувати аналіз, але й легко пояснити його результати іншим учасникам процесу. Це створює умови для впевненого прийняття рішень, які базуються на об’єктивних даних [8], [13].

Особливістю середовища є зручність роботи, оскільки весь аналітичний процес будується за допомогою перетягування елементів інтерфейсу. Такий підхід дозволяє швидко формувати нові сценарії аналізу, підготовлювати набори даних та здійснювати відбір ознак за критеріями інформативності без значних витрат часу. Завдяки цьому навіть користувачі без глибоких технічних навичок отримують можливість контролювати дані та розкривати їхній аналітичний потенціал [8], [13].

Veera активно використовується освітніми закладами, медичними установами та іншими організаціями для поєднання інформаційних потоків, створення звітності та підготовки матеріалів для прогнозних моделей. Результати аналізу можуть зберігатися у різних форматах або передаватися через корпоративні системи, електронні повідомлення та веб-середовище. Такий підхід робить програмне забезпечення універсальним інструментом для роботи з даними, що дозволяє організовувати їх відбір та підготовку з максимальною ефективністю [8], [13].

Основними характеристиками програмного забезпечення Rapid Insight Veera є такі [8], [13]:

– інтуїтивний інтерфейс: програмне забезпечення Rapid Insight Veera забезпечує простоту використання навіть для користувачів без досвіду програмування [8], [13];

– інтеграція даних: програмне забезпечення Rapid Insight Veera підтримує завантаження, обробку та поєднання інформації з різних джерел, включаючи бази даних, таблиці та хмарні сервіси [8], [13];

– візуальне моделювання процесів: програмне забезпечення Rapid Insight Veera дозволяє створювати сценарії аналізу даних у вигляді блок-схем без написання коду [8], [13];

– швидка підготовка даних: програмне забезпечення Rapid Insight Veera забезпечує очищення, трансформацію та нормалізацію даних для подальшого аналізу [8], [13];

– автоматизація завдань: програмне забезпечення Rapid Insight Veera дає можливість зберігати та повторно запускати процеси обробки даних користувача [8], [13];

– інтеграція з іншими інструментами: результати роботи програмного забезпечення Rapid Insight Veera можуть експортуватися у популярні статистичні пакети та системи візуалізації [8], [13];

– підтримка командної роботи: програмне забезпечення Rapid Insight Veera дозволяє кільком користувачам працювати над спільними проєктами та сценаріями обробки [8], [13];

– масштабованість: програмне забезпечення Rapid Insight Veera підходить як для невеликих організацій, так і для великих підприємств із великими масивами даних [8], [13];

– гнучкість аналізу: програмне забезпечення Rapid Insight Veera надає користувачам інструменти для дослідження даних, побудови прогнозів та створення моделей [8], [13];

– зниження вартості аналітики: програмне забезпечення Rapid Insight Veera дозволяє компаніям скорочувати витрати на впровадження складних програмних рішень завдяки простоті та універсальності [8], [13].

Серед недоліків програмного забезпечення Rapid Insight Veera можна відзначити обмежені можливості масштабування при роботі з дуже великими масивами даних, а також залежність від вбудованих алгоритмів, що іноді ускладнює гнучке налаштування процесів під специфічні завдання. Використання закритих механізмів обробки зменшує прозорість методів аналізу, що може бути критичним для досліджень, де необхідна повна інтерпретованість результатів. Додатковим обмеженням є складність інтеграції з деякими зовнішніми платформами та інструментами, що знижує універсальність рішення у складних багатокомпонентних проєктах [8], [13].

Програмне забезпечення Alteryx Analytics (рис. 1.2) призначене для об’єднання, аналізу та підготовки даних до подальшого використання у прогнозному моделюванні та прийнятті рішень. Його особливістю є інтуїтивно зрозумілий робочий процес, який дозволяє виконувати складні аналітичні завдання без залучення програмування. У середовищі Alteryx можна інтегрувати дані з внутрішніх джерел, зовнішніх сервісів та хмарних платформ, після чого вони обробляються з використанням методів статистичного аналізу, машинного навчання та просторової аналітики [8], [14].

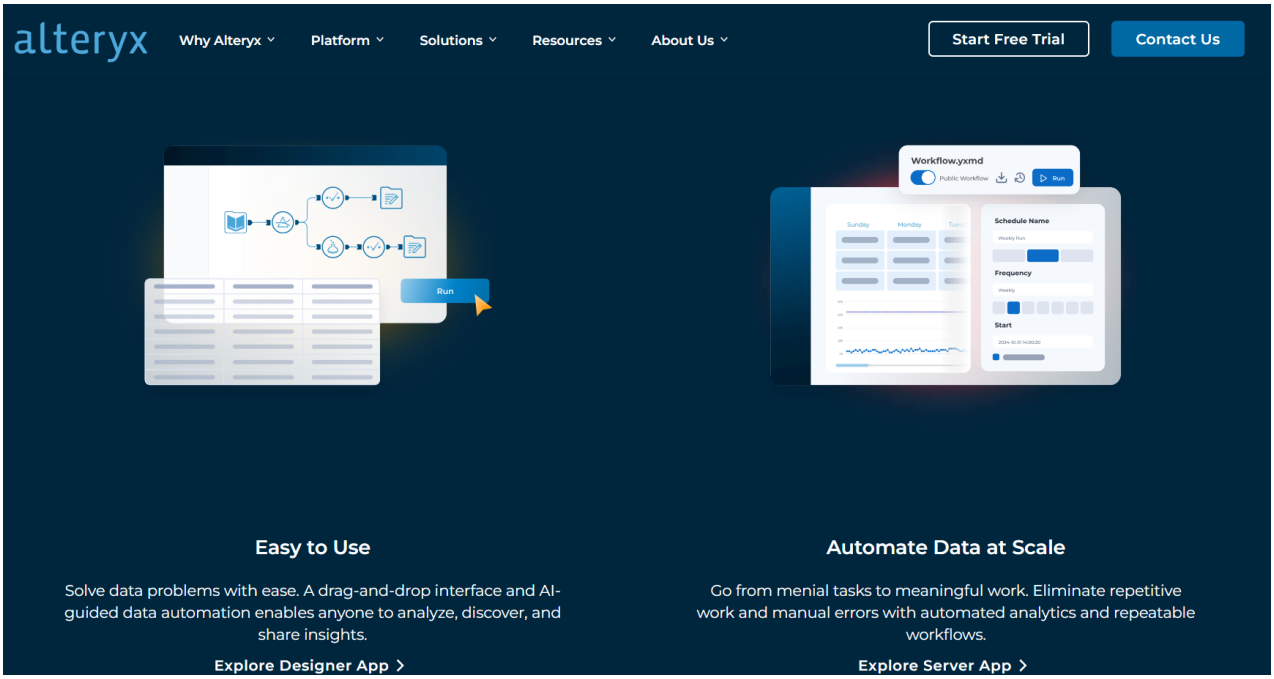


Рисунок 1.2 – Програмне забезпечення Alteryx Analytics [14]

Програмне забезпечення Alteryx Analytics підтримує побудову прогнозних моделей, у тому числі логістичної регресії, дерев рішень чи кластеризації, а також застосування технік зменшення розмірності, наприклад, аналізу головних компонент. Крім того, програмне забезпечення Alteryx Analytics дає можливість досліджувати дані за допомогою графічних візуалізацій і пошуку асоціативних зв'язків, що дозволяє отримувати глибші аналітичні висновки [8], [14].

Важливою функцією Alteryx Analytics є можливість відбору найбільш інформативних ознак, що робить аналітичні моделі більш точними та оптимізує обчислювальні ресурси. Процес відбору може здійснюватися на основі різних критеріїв та інтегруватися у загальний робочий процес підготовки даних, що особливо корисно при обробці великих і різнорідних масивів інформації [8], [14].

Alteryx Server забезпечує масштабованість і дозволяє автоматизувати виконання аналітичних процесів, планувати їх у часі та поширювати результати всередині організації. Alteryx Analytics Gallery надає можливість обміну аналітичними додатками та макросами, створюючи зручне середовище для колективної роботи. Завдяки цьому інструменту користувачі можуть швидко отримувати доступ до готових рішень, повторно використовувати їх та адаптувати під власні завдання [8], [14].

Завдяки поєднанню в одному середовищі підготовки, аналізу та відбору ознак, Alteryx Analytics дозволяє суттєво прискорити процес отримання цінних висновків, що робить його ефективним інструментом для бізнесу, науки та досліджень [8], [14].

Програмне забезпечення Alteryx Analytics має такі особливості [9], [13]:

– простота використання: інтерфейс програмного забезпечення Alteryx Analytics побудований на drag-and-drop принципах, що дозволяє працювати без необхідності програмування [8], [14];

– інтеграція даних: програмне забезпечення Alteryx Analytics підтримує з’єднання з великою кількістю джерел даних, включаючи бази даних, хмарні сервіси та локальні файли [8], [14];

– швидкість обробки: оптимізовані алгоритми забезпечують високу продуктивність при обробці великих обсягів інформації [8], [14];

– підготовка даних: програмне забезпечення Alteryx Analytics надає інструменти для очищення, трансформації та нормалізації даних [8], [14];

– аналітичні можливості: програмне забезпечення Alteryx Analytics надає вбудовані функції для статистичного аналізу, прогнозування та машинного навчання [8], [14];

– візуалізація процесів: робочі процеси відображаються у вигляді зрозумілих блок-схем [8], [14];

– автоматизація завдань: програмне забезпечення Alteryx Analytics надає можливість створення повторюваних сценаріїв для регулярного виконання процесів [8], [14];

– масштабованість: програмне забезпечення Alteryx Analytics підтримує роботи як на окремих комп’ютерах, так і у великих корпоративних середовищах [8], [14];

– колаборація: забезпечується спільна робота користувачів над аналітичними проєктами [8], [14];

– розширюваність: програмне забезпечення Alteryx Analytics підтримує можливість інтеграції з Python та R для складніших сценаріїв аналізу [8], [14].

Серед недоліків програмного забезпечення Alteryx Analytics можна відзначити високу вартість ліцензії, що обмежує його доступність для невеликих компаній та індивідуальних користувачів. Крім того, робота з великими масивами даних може вимагати значних обчислювальних ресурсів, що впливає на продуктивність системи. До недоліків також належить досить складний процес навчання, оскільки новим користувачам потрібно витратити багато часу на опанування широкого функціоналу та інтерфейсу [8], [14].

Програмне забезпечення Teradata (рис. 1.3) являє собою комплексне рішення для управління великими обсягами інформації та проведення глибокої аналітики. Його робота базується на багатохмарній платформі, яка поєднує у собі інструменти інтеграції, обробки та аналізу даних, забезпечуючи підприємствам високу гнучкість у розгортанні системи як у локальному середовищі, так і в хмарних сервісах. Платформа дозволяє працювати з даними у режимі реального часу, що робить можливим оперативне отримання результатів та прийняття рішень на основі актуальної інформації [9], [15].

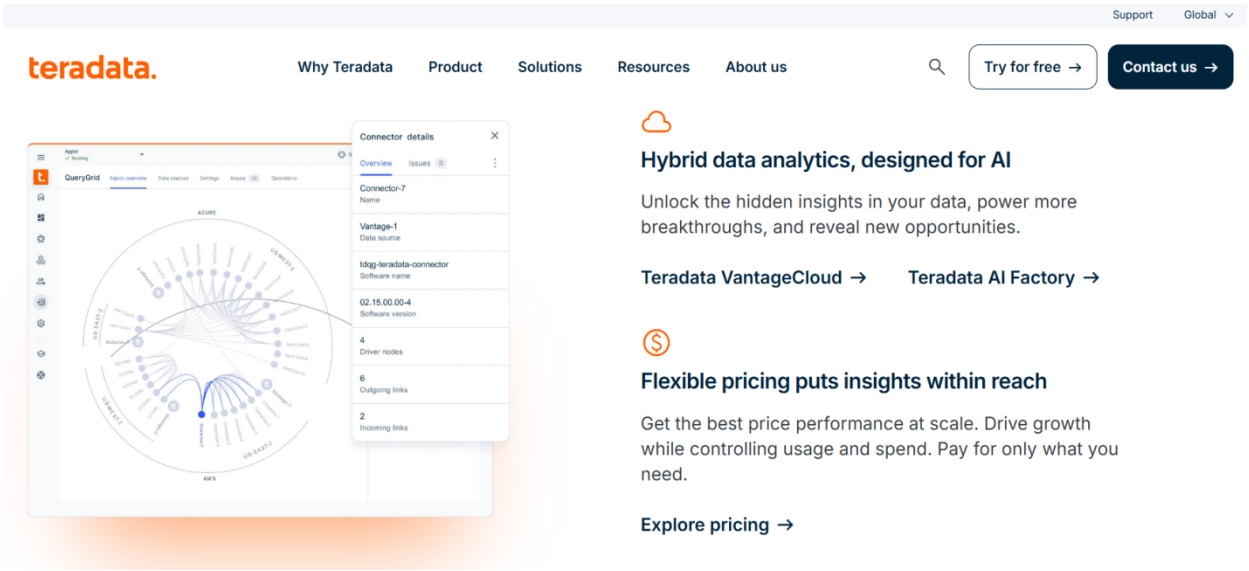


Рисунок 1.3 – Програмне забезпечення Teradata [15]

Особливою перевагою є можливість підтримки алгоритмів штучного інтелекту та методів машинного навчання, що дозволяє реалізовувати задачі прогнозування та виявляти закономірності у даних. Важливим функціоналом є відбір інформативних ознак, який здійснюється з урахуванням різних критеріїв якості, що дає змогу формувати оптимальні набори для побудови моделей. Завдяки цьому забезпечується підвищення точності аналітичних результатів і скорочення часу обробки [9], [15].

Програмне забезпечення підтримує роботу з різними форматами даних і забезпечує масштабованість на рівні корпоративних систем. Захищений доступ, гнучке управління ресурсами та прозора система контролю

використання роблять його ефективним інструментом для бізнесу, орієнтованого на аналіз та оптимізацію процесів [9], [15].

Програмне забезпечення Teradata має такі особливості [9], [15]:

– масштабованість: програмне забезпечення Teradata підтримує обробку дуже великих обсягів даних, забезпечуючи продуктивність навіть при зростанні навантаження [9], [15];

– архітектура: програмне забезпечення Teradata побудовано на основі паралельної обробки даних, що дозволяє ефективно виконувати складні запити [9], [15];

– оптимізація запитів: вбудований оптимізатор автоматично вибирає найефективніший план виконання запиту [9], [15];

– інтеграція: програмне забезпечення Teradata забезпечує з’єднання з різними джерелами даних, включаючи хмарні сервіси та корпоративні сховища [9], [15];

– безпека: реалізовані засоби аутентифікації, авторизації та захисту даних на рівні користувачів і транзакцій [9], [15];

– аналітичні можливості: програмне забезпечення Teradata підтримує складні аналітичні функції, включаючи OLAP-операції, машинне навчання та прогнозну аналітику [9], [15];

– надійність: висока відмовостійкість і механізми автоматичного відновлення забезпечують стабільну роботу системи [9], [15];

– інструменти адміністрування: програмне забезпечення Teradata надає зручні засоби моніторингу, керування продуктивністю та налаштування системи [9], [15];

– гнучкість розгортання: програмне забезпечення Teradata підтримує можливість встановлення у локальній інфраструктурі або використання у хмарних середовищах [9], [15];

– підтримка стандартів: програмне забезпечення Teradata підтримує сумісність з SQL та іншими промисловими стандартами спрощує інтеграцію з існуючими системами [9], [15].

Серед недоліків програмного забезпечення Teradata можна відзначити високу вартість ліцензування та підтримки, що робить його менш доступним для невеликих компаній. До того ж система вимагає значних апаратних ресурсів і високого рівня кваліфікації персоналу для налаштування та адміністрування. Обмеження у гнучкості масштабування порівняно з сучасними хмарними рішеннями також може стати проблемою для організацій, які прагнуть швидко адаптувати інфраструктуру під нові потреби. Крім того, оновлення та міграція даних у середовищі Teradata можуть бути складними та трудомісткими процесами [9], [15].

Порівняльну характеристику програмного забезпечення для аналізу даних та відбору ознак наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для аналізу даних та відбору ознак

Критерій порівняння	Rapid Insight Veera [13]	Alteryx Analytics [14]	Teradata [15]
Простота використання	+	+–	–
Швидкість обробки даних	+–	+	+
Масштабованість	–	+–	+
Можливості інтеграції з іншими системами	+–	+	+
Інструменти візуалізації	+–	+	+–
Аналітичні можливості	+–	+	+–

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для аналізу даних та відбору ознак. Проте деякі програмні засоби, що реалізують методи відбору ознак, попри свою практичну цінність, мають низку обмежень, які впливають на якість та ефективність їх використання. Складність деяких методів призводить

до значних обчислювальних витрат, що стає критичним при роботі з великими масивами даних. Крім того, реалізовані програмні рішення не завжди забезпечують достатню гнучкість, оскільки часто орієнтовані на використання певних підходів і можуть мати обмежені можливості налаштування для специфічних завдань. Ще однією суттєвою проблемою є ризик втрати важливої інформації, коли алгоритми відбору зменшують простір ознак надмірно агресивно, і, як наслідок, модель втрачає здатність до адекватного відображення складних залежностей. В окремих випадках труднощі викликає також інтерпретація результатів, оскільки автоматизовані системи не завжди надають зрозуміле пояснення того, чому були відкинуті певні характеристики. Обмеженість інтеграційних можливостей із зовнішніми аналітичними програмними платформами та недостатня оптимізація під конкретні архітектури обчислювальних систем також ускладнюють практичне застосування таких рішень. Тому актуальною є розробка програмного забезпечення, що реалізує методи відбору ознак.

При розробці програмного забезпечення, що реалізує методи відбору ознак, необхідно забезпечити такі функціональні вимоги:

– підтримка можливості завантаження вхідного набору даних;

– підтримка можливості виконання відбору ознак за допомогою різних методів;

– можливість отримання результатів відбору ознак, зокрема, набору інформативних ознак, значення загальної інформативності набору ознак та значення індивідуальної інформативності ознак;

– можливість збереження редукованої вибірки даних;

– підтримка можливості збереження результатів відбору ознак.

1.3 Висновки за розділом 1

Визначено, що відбір ознак є підходом до скорочення розмірності, що передбачає вибір обмеженої підмножини релевантних ознак з первинного

набору даних шляхом усунення шумових, надлишкових або малозначущих елементів. Виконання такого відбору здатне підвищити ефективність навчання моделей, забезпечити більшу точність прогнозування, зменшити обчислювальні витрати та полегшити інтерпретацію результатів. Відзначено, що надмірність даних у сучасних великих наборах змушує застосовувати розумні алгоритми для виявлення критично важливої інформації, що забезпечує ефективне формування моделей у задачах класифікації, регресії або кластеризації. Використання методів відбору ознак дозволяє скоротити обсяг даних для обробки, зменшити складність моделей, прискорити навчання та знизити ризик перенавчання. Проаналізовано методи відбору інформативних ознак.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для аналізу даних та відбору ознак. Проте деякі програмні засоби, що реалізують методи відбору ознак, попри свою практичну цінність, мають низку обмежень, які впливають на якість та ефективність їх використання. Складність деяких методів призводить до значних обчислювальних витрат, що стає критичним при роботі з великими масивами даних. Крім того, реалізовані програмні рішення не завжди забезпечують достатню гнучкість, оскільки часто орієнтовані на використання певних підходів і можуть мати обмежені можливості налаштування для специфічних завдань. Ще однією суттєвою проблемою є ризик втрати важливої інформації, коли алгоритми відбору зменшують простір ознак надмірно агресивно, і, як наслідок, модель втрачає здатність до адекватного відображення складних залежностей. В окремих випадках труднощі викликає також інтерпретація результатів, оскільки автоматизовані системи не завжди надають зрозуміле пояснення того, чому були відкинуті певні характеристики. Обмеженість інтеграційних можливостей із зовнішніми аналітичними програмними платформами та недостатня оптимізація під конкретні архітектури обчислювальних систем також ускладнюють практичне

застосування таких рішень. Тому актуальною є розробка програмного забезпечення, що реалізує методи відбору ознак.

Сформульовано функціональні вимоги до програмного забезпечення, що реалізує методи відбору ознак.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення, що реалізує методи відбору ознак, було використано мову програмування Python [16], [17].

Мова програмування Python належить до універсальних інструментів програмування, оскільки поєднує в собі можливості різних підходів, серед яких об’єктно-орієнтований, процедурний та функціональний. Така гнучкість надає змогу використовувати її як для створення невеликих експериментальних рішень, так і для розробки масштабних систем, що потребують поєднання різних методів. Наявність динамічної типізації та автоматичного управління пам’яттю спрощує процес розробки, що особливо важливо у випадках, коли потрібно швидко перевіряти наукові гіпотези та проводити експерименти [16], [17].

Значну роль у популярності Python відіграє її тісний зв’язок із галуззю штучного інтелекту та машинного навчання. Саме цією мовою створено переважну більшість сучасних інструментів для роботи з даними, серед яких TensorFlow, PyTorch, scikit-learn, NumPy та pandas, що сьогодні є основою практичного застосування методів аналізу даних. Розвинена система бібліотек для візуалізації, зокрема Matplotlib, Seaborn і Plotly, дає змогу швидко представити отримані результати у зрозумілій графічній формі [16], [17].

Важливим чинником розвитку мови програмування Python є активна міжнародна спільнота, що забезпечує регулярне оновлення екосистеми, створює відкриті ресурси й документацію, а також сприяє швидкому усуненню проблем. Така підтримка дозволяє Python залишатися одним із найдинамічніших інструментів сучасного програмування. Його застосування поширюється від веброзробки та бізнес-автоматизації до кібербезпеки, моделювання процесів і фінансових обчислень, що підтверджує універсальний характер мови [16], [17].

У результаті Python сприймається не лише як засіб написання програмного коду, а як цілісна екосистема, яка об’єднує різні галузі науки та промисловості, що й зумовлює її провідне становище серед мов програмування у світовій практиці [16], [17].

Застосування мови програмування Python у процесі розробки програмного забезпечення для реалізації методів відбору ознак визначається доцільним через поєднання властивостей, що відповідають потребам сучасних підходів до аналізу даних та побудови інтелектуальних систем. Важливим чинником є сформована екосистема бібліотек і фреймворків, яка охоплює інструменти для машинного навчання, статистичного моделювання та роботи з великими обсягами інформації. Завдяки цьому зменшується необхідність у самостійному створенні алгоритмів, а основна увага спрямовується на інтеграцію готових рішень, оптимізацію їх роботи та дослідження ефективності різних методів [16], [17].

Особливістю мови програмування Python є синтаксична простота, що робить мову зручною для швидкого створення прототипів і полегшує взаємодію між спеціалістами різного профілю. У випадках, коли в проєкті одночасно беруть участь дослідники даних і програмісти, зрозумілий стиль кодування сприяє ефективній співпраці. Можливість об’єднання завдань із попередньої обробки даних, побудови візуалізацій і впровадження алгоритмів у прикладні системи в межах одного середовища підтверджує універсальний характер мови програмування Python. Суттєве значення має також міжплатформеність та сумісність Python з іншими мовами й технологічними рішеннями, що відкриває широкі можливості для практичного використання створених програмних продуктів [16], [17].

Динамічний розвиток спільноти користувачів і безперервне вдосконалення бібліотек формують умови для реалізації гнучких і надійних рішень у сфері відбору ознак. Завдяки цьому Python розглядається як інструмент, що забезпечує оптимальний баланс між простотою застосування, продуктивністю та науковою обґрунтованістю підходів [16], [17].

Обґрунтування вибору мови програмування для реалізації засобів, що реалізує методи відбору ознак наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для створення засобів, що реалізує методи відбору ознак

Критерій порівняння мов програмування	Мова програмування		
	Python	C++	Java
Швидкість розробки прототипів	+	–	+–
Бібліотеки для машинного навчання та відбору ознак	+	+–	+–
Інтеграція з іншими інструментами	+	+–	+
Засоби візуалізації даних	+	–	+–
Популярність у науковій спільноті	+	+–	+–

Отже, для реалізації програмного забезпечення, що реалізує методи відбору ознак, обрано мову програмування Python, яка характеризується розвиненою екосистемою бібліотек і фреймворків, що забезпечують реалізацію алгоритмів машинного навчання, статистичного аналізу та роботи з великими масивами даних. Завдяки цьому знижується потреба у створенні алгоритмів з нуля, а розробка зосереджується на інтеграції, оптимізації та експериментуванні з різними методами. Крім того, мова Python відзначається високим рівнем зрозумілості синтаксису, що сприяє швидкому прототипуванню та спрощує процес командної роботи, коли над проєктом працюють як досвідчені програмісти, так і фахівці з даних, які не мають глибокого програмістського досвіду. Універсальність Python дозволяє поєднувати завдання з передобробки даних, візуалізації результатів та інтеграції моделей у прикладні системи в єдиному середовищі.

2.2 Вибір середовища розробки для створення програмного забезпечення, що реалізує методи відбору ознак

В якості середовища розробки для створення програмного забезпечення, що реалізує методи відбору ознак було обрано середовище PyCharm [18], [19].

Середовище розробки PyCharm характеризується високим рівнем гнучкості у налаштуванні робочого простору, що створює можливість адаптації інтерфейсу та інструментарію відповідно до специфіки завдань. Завдяки широкій системі плагінів середовище легко поєднується з додатковими технологіями, серед яких засоби для аналізу даних, візуалізації, веброзробки й роботи з хмарними платформами. Подібна універсальність робить його особливо корисним у сферах, де необхідно інтегрувати різні напрямки досліджень та технологічні підходи [18], [19].

Серед ключових характеристик варто відзначити потужний механізм тестування та перевірки якості коду. Вбудовані інструменти для автоматизованих тестів, контролю покриття та пошуку потенційних помилок дозволяють забезпечити високу надійність створених програмних рішень. У наукових і прикладних дослідженнях, де важливим є точне відтворення експериментів, ці можливості набувають особливого значення [18], [19].

Важливою складовою функціоналу є підтримка роботи з віддаленими серверами та контейнерними технологіями. Інтеграція з Docker і платформами для віддаленої розробки дає змогу виконувати обчислення великої складності на розподілених системах чи хмарних інфраструктурах без втрати зручності роботи в середовищі. Це відкриває перспективи для реалізації масштабованих методів аналізу, зокрема пов’язаних із відбором ознак у масивних наборах даних [18], [19].

У підсумку середовище розробки PyCharm постає не лише як стандартний засіб для програмування, а як комплексний інструмент, що об’єднує засоби розробки, тестування, оптимізації та розгортання,

забезпечуючи підвищення ефективності й надійності прикладних і наукових досліджень [18], [19].

Застосування середовища PyCharm у процесі створення програмного забезпечення для реалізації методів відбору ознак визначається його здатністю поєднувати широкий спектр інструментів, розроблених спеціально для ефективної роботи з мовою Python і бібліотеками обробки даних. Середовище забезпечує інтелектуальну підтримку під час написання коду, зручні засоби пересування між різними частинами проєкту та функції швидкої зміни його структури, що значно прискорює робочий процес і знижує ймовірність появи помилок [18], [19].

Особливе значення має підтримка роботи з віртуальними середовищами та менеджерами залежностей, що створює зручні умови для використання численних бібліотек машинного навчання й засобів аналізу даних. Інструменти для налагодження й профілювання середовища PyCharm дозволяють контролювати ефективність роботи алгоритмів і виконувати їх оптимізацію, що є особливо важливим у випадках обробки масштабних наборів даних [18], [19].

Додатково PyCharm надає розширені можливості для взаємодії з базами даних, містить засоби тестування та підтримує інтеграцію з системами керування версіями, що робить його придатним як для індивідуальних розробок, так і для командної роботи. Наявність функцій візуалізації безпосередньо у процесі програмування дає змогу дослідникові оперативно відстежувати результати експериментів та здійснювати їхній аналіз у реальному часі [18], [19].

У результаті використання цього середовища створюються сприятливі умови для підвищення ефективності роботи, зручності організації проєктів та забезпечення високої якості програмних продуктів, орієнтованих на реалізацію методів відбору ознак [18], [19].

Обґрунтування вибору середовища розробки для створення програмного забезпечення, що реалізує методи відбору ознак наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення, що реалізує методи відбору ознак

Критерій порівняння середовищ розробки	Середовища розробки		
	PyCharm	Jupyter Notebook	Spyder
Зручність написання та структурування проєктів ПЗ	+	–	+–
Інтелектуальне автодоповнення та рефакторинг	+	–	+
Робота з віртуальними середовищами та залежностями	+	+–	+–
Зручність для аналізу та візуалізації даних	+	+	+–
Засоби налагодження та профілювання	+	–	+–

Таким чином, для створення програмного забезпечення, що реалізує методи відбору ознак обрано середовище розробки PyCharm завдяки поєднанню інструментів, орієнтованих на ефективну роботу з мовою Python та бібліотеками для аналізу даних. PyCharm забезпечує інтелектуальне автодоповнення коду, зручну навігацію по проєкту та можливості швидкого рефакторингу, що істотно прискорює процес розробки й мінімізує кількість помилок. Важливою перевагою є інтеграція з віртуальними середовищами та

системами керування залежностями, що дозволяє легко організувати роботу з різними бібліотеками машинного навчання та інструментами для обробки даних. PyCharm підтримує зручні засоби налагодження та профілювання, що є необхідними при оптимізації алгоритмів відбору ознак, особливо коли обробляються великі масиви інформації.

2.3 Розробка методу відбору ознак

Необхідність створення методу відбору ознак зумовлена складністю та багатогранністю процесів аналізу даних, розробки програмного забезпечення та обробки інформації, де традиційні підходи часто не забезпечують належного рівня точності, швидкості або зручності. Існуючі програмні інструменти дають певні результати, але їх ефективність зазвичай обмежена через недосконалу структуру, недостатню інтеграцію сучасних алгоритмів або низьку гнучкість у роботі з великими обсягами інформації. Саме тому виникає потреба у новому методі, який би дозволив поєднати різні етапи роботи в єдиний логічний процес та підвищити якість кінцевого результату.

Одним із можливих способів підвищення ефективності відбору інформативних ознак є використання комбінованого підходу, у якому поєднано кілька різних критеріїв. На початковому етапі з вихідного набору усуваються атрибути, значення яких залишаються незмінними, що дає змогу уникнути включення до моделі ознак із нульовою дисперсією. Після цього для решти характеристик обчислюються показники статистичних тестів, що дозволяє оцінити їхню значущість відносно цільової змінної.

Додатково виконується аналіз взаємної інформації між ознаками та вихідним параметром, що відкриває можливість оцінити ступінь залежності та відокремити найбільш релевантні атрибути. На основі отриманих значень визначаються межі, в межах яких показники взаємної інформації вважаються прийнятними. Далі кожна характеристика аналізується у порівнянні з іншими, і якщо виявляється, що вона перебуває у сильній залежності з іншою ознакою,

водночас має нижчі значення за допоміжними критеріями та демонструє слабший зв’язок із цільовою змінною, така характеристика підлягає вилученню.

У результаті формується підмножина незалежних та найбільш інформативних ознак, що забезпечує оптимальний баланс між кількістю параметрів і якістю побудови моделі. Запропонований метод відрізняється тим, що враховує не лише статистичну значущість та дисперсійні властивості, а й взаємозалежність між окремими атрибутами, що дозволяє знизити надмірність та підвищити стійкість кінцевих результатів.

Графічну інтерпретацію запропонованого методу відбору ознак подамо за допомогою рис. 2.1.

Рисунок 2.1 – Графічна інтерпретація методу відбору ознак

Розроблений метод передбачає кілька послідовних етапів. На першому етапі здійснюється підготовка вхідних даних або умов задачі. Це включає перевірку наявної інформації, її очищення від шумів, відсутніх чи некоректних значень, а також приведення до уніфікованого формату. Завдяки цьому забезпечується єдність структури та уникнення помилок у подальшій обробці.

Другий етап полягає у визначенні ключових характеристик або параметрів, які мають найбільший вплив на кінцевий результат. Для цього застосовуються методи відбору ознак або попереднього аналізу, що дозволяє зменшити обсяг обчислень і водночас підвищити точність. Виділення суттєвих характеристик допомагає зосередитись лише на найбільш значущих аспектах задачі.

Третій етап охоплює власне застосування основного алгоритму, який реалізує запропонований метод. У цьому процесі відбувається поступова обробка даних відповідно до визначених правил чи моделей. Алгоритм побудований так, щоб враховувати як локальні, так і глобальні залежності, що забезпечує більш глибоке розуміння структури інформації та дозволяє отримати точніший результат.

На четвертому етапі відбувається перевірка правильності та ефективності отриманих результатів. Для цього застосовуються методи оцінки, наприклад, обчислення похибок, порівняння з еталонними значеннями або перевірка на тестових прикладах. За необхідності проводиться корекція параметрів, що дозволяє удосконалити роботу методу та зробити його більш адаптивним.

П’ятий етап передбачає інтерпретацію результатів. Це означає не лише представлення кінцевих значень, а й пояснення їхнього змісту, формування висновків і рекомендацій. Саме на цьому етапі метод набуває практичної цінності, оскільки перетворює сухі дані чи проміжні обчислення на зрозумілу інформацію, яка може бути використана для прийняття рішень або побудови прогнозів.

Таким чином, запропоновано метод відбору ознак, який полягає у поступовій підготовці даних, виділенні ключових характеристик, застосуванні адаптивного алгоритму, перевірці отриманих результатів та їх інтерпретації, що дозволяє підвищити точність аналізу, зменшити вплив випадкових похибок та забезпечити більшу практичну значущість кінцевих висновків.

2.4 Висновки за розділом 2

Для реалізації програмного забезпечення, що реалізує методи відбору ознак обрано мову програмування Python, яка характеризується розвиненою екосистемою бібліотек і фреймворків, що забезпечують реалізацію алгоритмів машинного навчання, статистичного аналізу та роботи з великими масивами даних. Завдяки цьому знижується потреба у створенні алгоритмів з нуля, а розробка зосереджується на інтеграції, оптимізації та експериментуванні з різними методами. Крім того, мова Python відзначається високим рівнем зрозумілості синтаксису, що сприяє швидкому прототипуванню та спрощує процес командної роботи, коли над проєктом працюють як досвідчені програмісти, так і фахівці з даних, які не мають глибокого програмістського досвіду. Універсальність Python дозволяє поєднувати завдання з передобробки даних, візуалізації результатів та інтеграції моделей у прикладні системи в єдиному середовищі.

Для створення програмного забезпечення, що реалізує методи відбору ознак обрано середовище розробки PyCharm завдяки поєднанню інструментів, орієнтованих на ефективну роботу з мовою Python та бібліотеками для аналізу даних. PyCharm забезпечує інтелектуальне автодоповнення коду, зручну навігацію по проєкту та можливості швидкого рефакторингу, що істотно прискорює процес розробки й мінімізує кількість помилок. Важливою перевагою є інтеграція з віртуальними середовищами та системами керування залежностями, що дозволяє легко організувати роботу з різними бібліотеками машинного навчання та інструментами для обробки даних. PyCharm

підтримує зручні засоби налагодження та профілювання, що є необхідними при оптимізації алгоритмів відбору ознак, особливо коли обробляються великі масиви інформації.

Створено метод відбору ознак, який полягає у поступовій підготовці даних, виділенні ключових характеристик, застосуванні адаптивного алгоритму, перевірці отриманих результатів та їх інтерпретації, що дозволяє підвищити точність аналізу, зменшити вплив випадкових похибок та забезпечити більшу практичну значущість кінцевих висновків.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення, що реалізує методи відбору ознак

Структуру програмного забезпечення, що реалізує методи відбору ознак наведено на рис. 3.1.

Рисунок 3.1 – Структура програмного забезпечення,
що реалізує методи відбору ознак

Як видно з рис. 3.1, структура програмного забезпечення, що реалізує методи відбору ознак, організована у вигляді п’яти взаємопов’язаних модулів: модуль завантаження та підготовки даних, модуль управління ознаками,

модуль методів відбору, модуль оцінки якості відібраних ознак, модуль візуалізації результатів.

Модуль завантаження та підготовки даних відповідає за отримання інформації з різних джерел, перевірку її якості, заповнення пропусків і приведення до уніфікованого формату. Він забезпечує очищення даних, нормалізацію та створення вхідних матриць ознак.

Модуль управління ознаками забезпечує попередній аналіз простору ознак, їх кодування, трансформація та усунення надлишковості. Цей компонент дозволяє виконувати масштабування та приведення ознак до єдиного масштабу для подальшого аналізу.

Модуль методів відбору містить реалізації різних підходів до вибору релевантних характеристик. У його складі знаходяться реалізації алгоритмів фільтраційного відбору, та інші методи відбору ознак. Завдяки цьому забезпечується гнучкість у виборі стратегії аналізу.

Модуль оцінки якості відібраних ознак: його функції полягають у використанні різних метрик продуктивності, перехресної перевірки та аналізу стабільності наборів ознак. Він дає змогу визначити, наскільки ефективно вибрані характеристики впливають на точність моделі.

Модуль візуалізації результатів забезпечує графічне відображення процесів та підсумкових наборів ознак. У ньому реалізуються діаграми важливості характеристик, графіки точності моделей при різних підмножинах ознак, а також інші форми представлення результатів для зручності інтерпретації. Цей модуль також дозволяє експортувати відібрані ознаки у стандартних форматах, інтегрувати їх у моделі машинного навчання та формувати звітність для подальшого використання в аналітичних завданнях.

3.2 Функціонування ПЗ, що реалізує методи відбору ознак

Програмне забезпечення, що застосовує методи відбору ознак, функціонує як інструмент оптимізації інформаційного простору даних. Його

робота спрямована на виявлення найбільш значущих характеристик, які визначають якість моделей аналізу та прогнозування. У процесі функціонування спочатку здійснюється підготовка вхідних даних, після чого активуються алгоритмічні процедури, здатні оцінювати внесок кожної змінної у загальну результативність. Оцінювання може базуватися на статистичних показниках, навчанні моделей чи пошуку комбінацій, що мінімізують надлишковість та шум.

Поступово простір ознак звужується, і програмне забезпечення формує підмножину характеристик, яка найкраще описує сутність досліджуваного явища. У результаті моделі, що будуються на основі відібраних змінних, демонструють кращу узагальнювальну здатність, підвищену точність і зменшену обчислювальну складність. Така програмна система працює як фільтр, що відсіює неінформативні або надлишкові дані, залишаючи лише те, що суттєво впливає на результат.

Функціонування програмного забезпечення, що реалізує методи відбору ознак, подамо за допомогою схеми, зображеної на рис. 3.2.

Проаналізуємо більш детально етапи функціонування програмного забезпечення, що реалізує методи відбору ознак. На початку здійснюється отримання даних із визначених джерел та перевірка доступності кожного ресурсу. Дані імпортуються у внутрішній формат, який підтримує як таблиці, так і напівструктуровані записи. Паралельно фіксується інформація про походження, час формування вибірки та параметри з’єднань, щоб забезпечити відтворюваність. Виявляються базові характеристики масиву, зокрема кількість спостережень, наявність текстових полів, часових міток та категоріальних значень. Визначаються ключі з’єднання у випадках, коли дані надходять з кількох таблиць, і перевіряється коректність об’єднання без втрати записів.

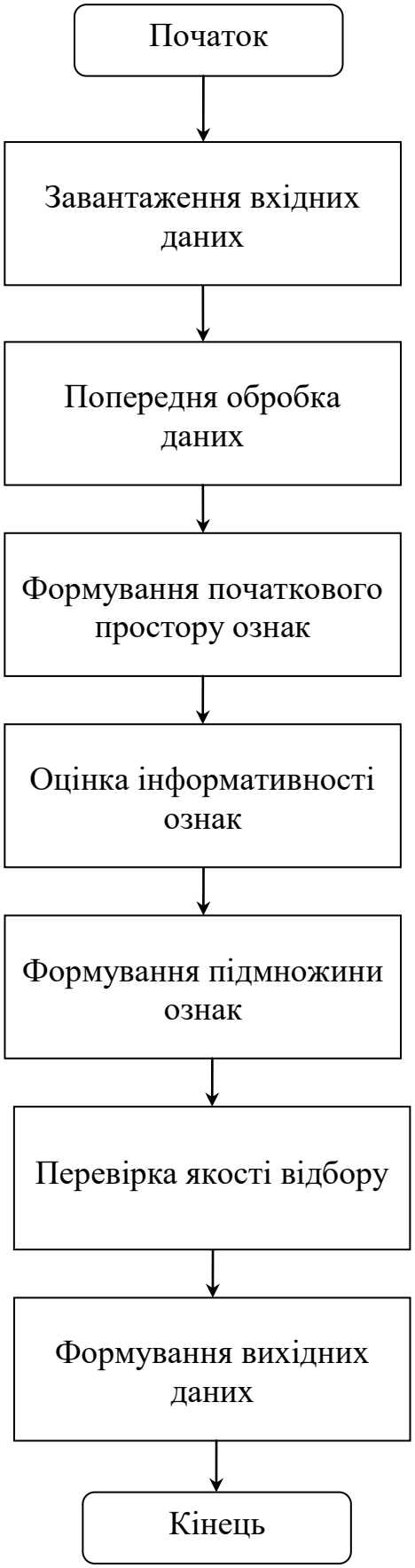


Рисунок 3.2 – Схема функціонування програмного забезпечення, що реалізує методи відбору ознак

Далі виконується підготовка даних, у межах якої усуваються проблеми якості. Пропуски обробляються узгодженим підходом, що враховує тип ознаки та характер розподілу, аномалії виявляються за допомогою статистичних та модельних процедур і за потреби коригуються або позначаються. Категоріальні поля кодуються методами, сумісними з обраними моделями, числові змінні нормалізуються або стандартизуються з урахуванням подальшої перевірки на валідаційних підвибірках. Проводиться синхронізація часових рядів, якщо ознаки утворені подіями з різною частотою, і виконується вирівнювання часових вікон, щоб уникнути потрапляння майбутньої інформації у навчальний етап.

Після очищення формується початковий простір ознак. На цьому кроці уточнюється ролева структура змінних, де одні поля визначаються як предиктори, інші як ціль або службові ідентифікатори. Усунення дубльованих або детерміновано похідних характеристик дозволяє зменшити надлишковість. Виявляються константні та майже константні стовпці, що не несуть корисного сигналу для навчання. Виконується попередній огляд кореляцій та нелінійних залежностей, щоб зрозуміти, чи є групи ознак зі схожою поведінкою, які вимагають спеціального поводження під час відбору.

Далі відбувається вибір стратегії відбору. Розглядаються підходи, орієнтовані на незалежні від моделі критерії, методи, що обгортають процедуру навчання, а також варіанти, де важливості визначаються параметрами моделі або штрафами. Рішення приймається з урахуванням розміру даних, очікуваної форми зв’язку між ознаками та ціллю, допустимих обчислювальних витрат і вимог до інтерпретованості. За необхідності комбінуються кілька механізмів у каскад, де прості швидкі фільтри використовуються як попередній бар’єр, а більш витратні процедури застосовуються до звуженої підмножини.

Після визначення стратегії проводиться оцінювання інформативності. Для фільтраційних підходів використовується порівняння розподілів цільової змінної за рівнями предикторів або аналіз асоціацій між числовими

величинами. У випадках обгорток моделювання виконується у циклі з повторною перевіркою, а показники якості реєструються для кожного кандидата. За вбудованих методів коефіцієнти або ваги, отримані під час навчання, інтерпретуються як міра корисності. На цьому етапі контролюється відсутність витоку даних, тому перетворення, які залежать від розподілу, налаштовуються виключно на навчальних підвибірках.

Після обчислення важливостей формується підмножина ознак. Застосовується порогування, ранжування або покрокові процедури додавання та вилучення за узгодженими правилами. У разі сильно корельованих предикторів зберігається представник групи, що має більш стабільний зв’язок із ціллю або кращу поведінку на валідаційних фолдах. Якщо передбачено галузеві обмеження, пріоритет надається змінним, які мають зрозумілий фізичний або бізнесовий сенс. У підсумку утворюється компактний набір, здатний зберігати предиктивну спроможність без зайвої складності.

Далі перевіряється якість відбору на незалежних підвибірках. Навчальна процедура виконується на звуженому просторі ознак, результати оцінюються на відкладених даних або за допомогою перехресної перевірки. Вимірюється стабільність вибору, щоб переконатися, що різні випадкові розбиття не призводять до радикально інших рішень. Проводиться порівняння з базовою моделлю на повному наборі або на альтернативних підмножинах, щоб підтвердити, що скорочення простору не погіршує узагальнювальну здатність і справді зменшує обчислювальні витрати.

Після первинної валідації виконується тонке налаштування. Параметри порогів, штрафів і кроків пошуку коригуються з урахуванням компромісу між точністю, інтерпретованістю та швидкодією. За необхідності повторно застосовуються попередні стадії до підмножини кандидатів, зокрема з урахуванням нових інженерних перетворень, що покращують роздільну здатність моделі. Особлива увага приділяється відтворюваності: фіксуються випадкові стани, зберігаються конфігурації та версії перетворень, щоб будь-який крок можна було відтворити.

На завершення результати узгоджуються з вимогами інтеграції. Відібрані ознаки експортуються у стандартизованому вигляді разом з описом перетворень, залежностей і версіями артефактів. Готуються підсумкові звіти з поясненнями причин включення та виключення змінних, з наведенням показників стабільності та чутливості. Формуються рекомендації щодо подальшої підтримки, зокрема періодичної переоцінки набору ознак у разі зміни розподілів даних або появи нових джерел.

3.3 Особливості реалізації програмного забезпечення для відбору ознак

Реалізація програмного забезпечення для відбору ознак дозволила розробити декілька функцій, які забезпечують автоматизацію процесу аналізу та збереження найбільш інформативних характеристик даних. Усі функції програмного забезпечення, що реалізує методи відбору ознак, працюють у комплексі, забезпечуючи ефективне скорочення розмірності даних, підвищення швидкодії алгоритмів та збереження найціннішої інформації для подальшого аналізу чи побудови прогнозних моделей.

У ході побудови програмної системи створено набір функцій, що дозволяють зчитувати початкові дані, перетворювати їх у зручний для подальшої обробки формат та застосовувати алгоритмічні методи для оцінювання значущості кожної ознаки. Програмне забезпечення для відбору ознак містить функції, які відповідають за нормалізацію даних, перевірку їх на наявність пропусків та викидів, а також за обчислення показників важливості ознак за різними методиками. Однією з функцій передбачено відбір параметрів на основі їх взаємозв’язку з цільовою змінною, деякі інші орієнтовані на усунення надлишкових характеристик, що мають високий ступінь кореляції між собою. Також реалізовано механізм, який дозволяє виконувати оцінювання інформативності ознак у межах побудованої моделі,

де найбільш значущі з них виділяються для подальшого використання, а менш корисні відсіюються.

Основна функція, що відповідає за попередню оцінку даних, у своїй роботі зосереджується на виявленні надлишкових та неінформативних змінних. Усередині неї реалізується механізм розрахунку кореляційних залежностей між ознаками, де для кожної пари обчислюється показник взаємозв'язку. Якщо виявляється сильна залежність, то одна з ознак позначається як кандидат на вилучення. При цьому враховується також взаємозв'язок із цільовою змінною, щоб не втратити важливу інформацію.

Інша функція зосереджує свою логіку на використанні статистичних критеріїв. Вона працює за принципом поступового обчислення значущості кожної ознаки. Всередині відбувається розрахунок показників, що відображають рівень впливу змінної на результат. Якщо отримане значення не перевищує порогу, закладеного у програму, ознака позначається як малозначуща. Для цього використовується серія тестів, що дозволяє уникнути випадкових відхилень та підвищити надійність результату.

Додатково функції для оптимізації простору ознак реалізують внутрішню логіку, пов'язану з відбором підмножини характеристик. Усередині закладено механізм, що перебирає можливі комбінації змінних і обчислює показники якості моделі при кожному варіанті. На цьому етапі застосовується принцип поступового вилучення або додавання ознак, що дозволяє сформувати оптимальний набір. Внутрішня логіка такої функції працює циклічно, поступово звужуючи простір до найбільш придатних характеристик.

Окремий блок функцій відповідає за роботу з алгоритмами машинного навчання. У цьому випадку внутрішня логіка полягає в навчанні моделі з повним набором ознак і подальшій оцінці важливості кожної з них. Для цього застосовуються спеціальні механізми, які зберігають інформацію про вагові коефіцієнти або про вплив змінних на точність передбачення. Функція

автоматично порівнює отримані результати, після чого визначає найвагоміші характеристики.

Усі функції працюють у взаємозв’язку, оскільки одна передає результати іншій. Внутрішня логіка їх поєднання полягає у поступовому фільтруванні, статистичній перевірці та оптимізації простору ознак. У результаті формується компактний набір характеристик, що зберігає найважливішу інформацію для побудови моделей і при цьому зменшує обчислювальні витрати.

3.4 Проєктування інтерфейсу програмного забезпечення, що реалізує методи відбору ознак

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням, що реалізує методи відбору ознак виконано з урахуванням вимог технічного завдання. При розробленні програмного забезпечення, що реалізує методи відбору ознак враховані функціональні вимоги до програми:

– підтримка можливості завантаження вхідного набору даних;

– підтримка можливості виконання відбору ознак за допомогою різних методів;

– можливість отримання результатів відбору ознак, зокрема, набору інформативних ознак, значення загальної інформативності набору ознак та значення індивідуальної інформативності ознак;

– можливість збереження редукованої вибірки даних;

– підтримка можливості збереження результатів відбору ознак.

Інтерфейс головної форми програмного забезпечення, що реалізує методи відбору ознак наведено на рис. 3.3.

Екранну форму з результатом відбору інформативних ознак наведено на рис. 3.4.

Рисунок 3.3 – Інтерфейс головної форми програмного забезпечення, що реалізує методи відбору ознак

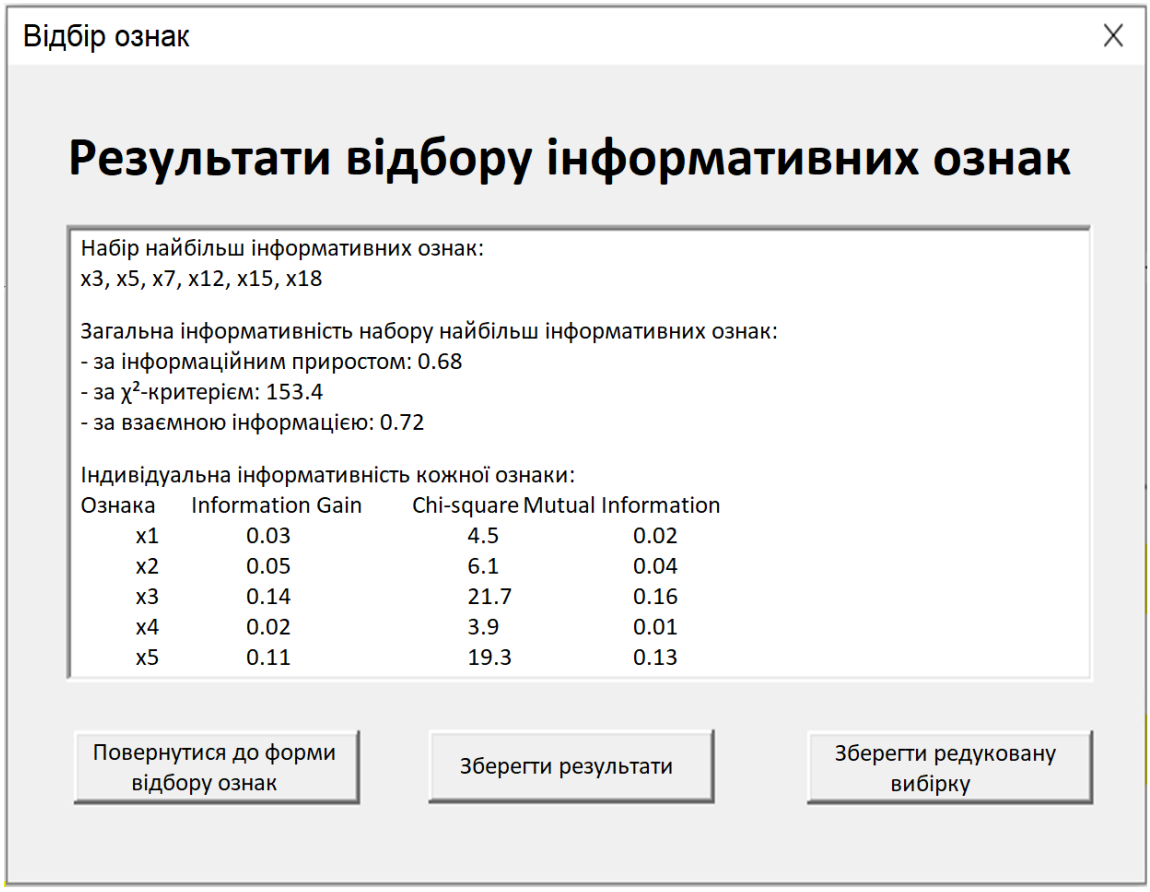


Рисунок 3.4 – Екранна форма з результатом відбору інформативних ознак

3.5 Висновки за розділом 3

Розроблено програмне забезпечення, що реалізує методи відбору ознак.

Запропоновано структуру програмного забезпечення, що реалізує методи відбору ознак. Визначено, що структура програмного забезпечення, що реалізує методи відбору ознак, організована у вигляді п’яти взаємопов’язаних модулів: модуль завантаження та підготовки даних, модуль управління ознаками, модуль методів відбору, модуль оцінки якості відібраних ознак, модуль візуалізації результатів.

Описано функціонування програмного забезпечення, що реалізує методи відбору ознак. Визначено, що у процесі функціонування програми спочатку здійснюється підготовка вхідних даних, після чого активуються алгоритмічні процедури, здатні оцінювати внесок кожної змінної у загальну результативність. Оцінювання може базуватися на статистичних показниках, навчанні моделей чи пошуку комбінацій, що мінімізують надлишковість та шум, поступово простір ознак звужується, і програмне забезпечення формує підмножину характеристик, яка найкраще описує сутність досліджуваного явища.

Описано особливості реалізації програмного забезпечення, що реалізує методи відбору ознак. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням, що реалізує методи відбору ознак.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для відбору інформативних ознак. Програма написана на мові Python, яка характеризується розвиненою екосистемою бібліотек і фреймворків, що забезпечують реалізацію алгоритмів машинного навчання, статистичного аналізу та роботи з великими масивами даних. В якості середовища розробки для створення програмного забезпечення, що реалізує методи відбору ознак обрано PyCharm завдяки поєднанню інструментів, орієнтованих на ефективну роботу з мовою Python та бібліотеками для аналізу даних.

Програмне забезпечення, що реалізує методи відбору ознак, забезпечує виконання таких функцій:

– підтримка можливості завантаження вхідного набору даних;

– підтримка можливості виконання відбору ознак за допомогою різних методів;

– можливість отримання результатів відбору ознак, зокрема, набору інформативних ознак, значення загальної інформативності набору ознак та значення індивідуальної інформативності ознак;

– можливість збереження редукованої вибірки даних;

– підтримка можливості збереження результатів відбору ознак.

Для експлуатації програмного забезпечення, що реалізує методи відбору ознак, необхідно таке апаратне та програмне забезпечення:

– на апаратному рівні необхідно, щоб центральний процесор мав тактову частоту не нижче 4 ГГц, оскільки виконання алгоритмів відбору ознак передбачає багатократні ітерації та обчислення статистичних характеристик. Рекомендується використання багатоядерного процесора, що дозволить прискорити паралельну обробку даних. Обсяг оперативної пам’яті має бути достатнім для завантаження вибірок та проміжних результатів під час

обчислень, тому мінімально необхідно чотири гігабайти, а оптимально – від восьми та вище, особливо при роботі з великими наборами даних. Для зберігання результатів експериментів та проміжних файлів рекомендується наявність жорсткого диска або твердотільного накопичувача з обсягом пам’яті не менше п’ятидесяти гігабайтів;

– на апаратному рівні необхідна операційна система. Оптимально застосовувати операційні системи сімейства Linux або Windows останніх версій. Потрібно мати встановлене програмне забезпечення для роботи з мовою Python, а також спеціалізовані бібліотеки для машинного навчання та аналізу даних, такі як NumPy, Pandas, Scikit-learn, TensorFlow.

Функціональні характеристики розробленого програмного забезпечення, що реалізує методи відбору ознак, наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення, що реалізує методи відбору, ознак наведено у додатку Б.

4.2 Характеристики програми, що реалізує методи відбору ознак

Програмне забезпечення, яке реалізує методи відбору ознак, характеризується здатністю автоматизувати процес визначення найбільш інформативних параметрів у даних. Його функціонування базується на аналізі значущості змінних з урахуванням їх впливу на точність моделей. У роботі програмного забезпечення передбачено використання статистичних критеріїв, методів на основі моделей та алгоритмів оптимізації, що дозволяють скорочувати кількість ознак без втрати інформативності.

Програмне забезпечення, яке реалізує методи відбору ознак, підтримує різні стратегії відбору. Важливою характеристикою є також можливість інтеграції з інструментами попередньої обробки та побудови моделей, завдяки чому відбір ознак здійснюється як частина єдиного циклу аналізу. Робота програмного забезпечення, яке реалізує методи відбору ознак, спрямована на

підвищення ефективності моделей, зменшення обчислювальних витрат і зниження ризику перенавчання.

Додатково забезпечується можливість оцінювання якості відбору шляхом порівняння результатів побудованих моделей із використанням повного та скороченого набору ознак. Це створює умови для прийняття рішень щодо балансу між точністю прогнозування і швидкістю обчислень. Таким чином, програмне забезпечення, яке реалізує методи відбору ознак, виступає важливим інструментом у завданнях інтелектуального аналізу даних і машинного навчання, формуючи основу для побудови надійних і стійких моделей.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення, що реалізує методи відбору ознак необхідно запустити відповідний файл, після чого необхідно завантажити вхідний набір даних, задати параметри відбору та ініціювати обчислювальний процес.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення, що реалізує методи відбору ознак, є набір даних, що складається з множини змінних та цільової ознаки, яка визначає залежність у досліджуваній задачі. До вхідних даних також можуть належати параметри налаштувань, що задають типи методів відбору, критерії оцінювання та порогові значення, які впливають на процес формування підмножини ознак.

Вихідними даними програмного забезпечення, що реалізує методи відбору ознак, є скорочений набір змінних, що вважаються найбільш

інформативними для побудови моделі, а також результати їх оцінювання за допомогою вибраних критеріїв.

4.3.3 Повідомлення

Користувач програмного забезпечення, що реалізує методи відбору ознак, може отримати такі повідомлення: повідомлення про завершення окремих етапів обчислень, попередження про відсутність достатньої варіативності у даних чи надмірну кореляцію між ознаками, а також повідомлення про помилки, які можуть виникати у разі некоректного формату даних або відсутності необхідних параметрів.

4.4 Виконання програмного забезпечення, що реалізує методи відбору ознак

Програмне забезпечення, що реалізує методи відбору ознак, створено у вигляді програмних модулів для можливості подальшої інтеграції з іншими програмними системами аналізу даних, прогнозування, розпізнавання тощо.

Крім того, реалізовано графічний інтерфейс для роботи кінцевих користувачів, які не є фахівцями в галузі програмування.

Після запуску програмного забезпечення, що реалізує методи відбору ознак, користувачу відображається головна форма (рис. 4.1). Тут необхідно завантажити вибірку вхідних даних шляхом натискання кнопки Завантажити дані. Потім можна провести відбір інформативних ознак одним з методів: відбір ознак з використанням фільтраційного підходу (оцінювання індивідуальної значущості ознак та вибору тих ознак, які характеризуються максимальними оцінками індивідуальної інформативності), відбір з послідовним додаванням ознак, відбір з послідовним видаленням ознак, метод повного перебору, відбір ознак за допомогою генетичного алгоритму та відбір ознак з використанням запропонованого методу.

Рисунок 4.1 – Інтерфейс головної форми програмного забезпечення, що реалізує методи відбору ознак

Після вибору методу відбору ознак та натискання відповідної кнопку користувачу відображається форма з результатами відбору ознак (рис. 4.2). Тут виводиться набір найбільш інформативних ознак, загальна інформативність відібраного набору, обчислена за допомогою різних підходів, а також інформація стосовно індивідуальної інформативності кожної з ознак у базовому наборі даних.

Користувач має можливість зберегти редуковану вибірку даних, тобто вибірку яка містить для кожного екземпляру лише значення відібраних інформативних ознак.

Такі вибірки даних у подальшому можуть застосовуватися для синтезу розпізнавальних або прогностичних моделей, які будуть надійними та стійкими, оскільки будуть побудовані без використання значень неінформативних та надлишкових ознак.

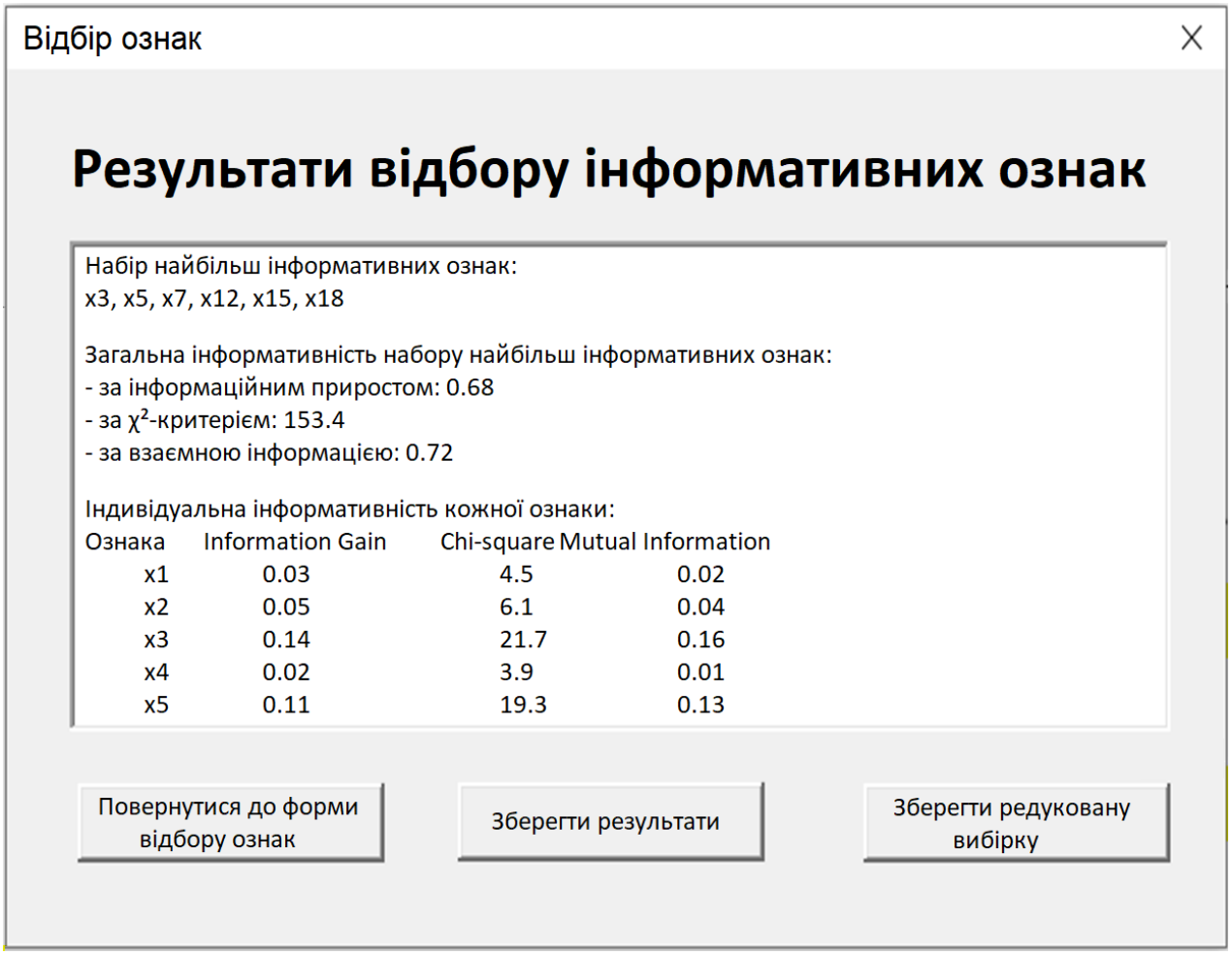


Рисунок 4.2 – Форма з результатом відбору інформативних ознак

За допомогою розроблених програмних модулів було досліджено різні підходи до відбору інформативних ознак. Вхідними даними для проведення досліджень слугував набір з більш, ніж 8000 екземплярів, у якому подано опис грибів за 22 зовнішніми характеристиками. Для кожного екземпляра визначено належність до класу їстівних або отруйних. Масив містив велику кількість спостережень і був охарактеризований множиною ознак, серед яких одна використовувалася як цільова та відображала токсичність. Приклад вихідної структури даних продемонстровано на ілюстрації.

Методика експериментів будувалася на зіставленні продуктивності традиційних підходів до відбору ознак із розробленим методом. У процесі роботи оцінювалися ключові характеристики, зокрема витрати часу на відбір, швидкість подальшого навчання класифікаційної моделі та отримана точність

прогнозування. Для стандартних технік застосовувались критерії F-значення, взаємної інформації та хі-квадрат. На початковому етапі проводилося обчислення цих показників для всіх атрибутів, після чого за допомогою відповідних інструментів було сформовано підмножину найбільш значущих ознак. Окремо випробовувався підхід, що ґрунтується на відсіві ознак із нульовою варіативністю значень, який реалізовувався через метод з параметрами за замовчуванням. Як модель класифікації використовувався алгоритм наївного байєсівського класифікатора у варіанті з гаусівським розподілом, що входить до складу відомої бібліотеки машинного навчання.

Результати перевірки було представлено у вигляді графічних матеріалів. На окремих діаграмах показано залежність значень обчислених критеріїв від відповідних ознак.

На рис. 4.3 наведено значення F-критерію як оцінки інформативності кожної з ознак у навчальному наборі даних.

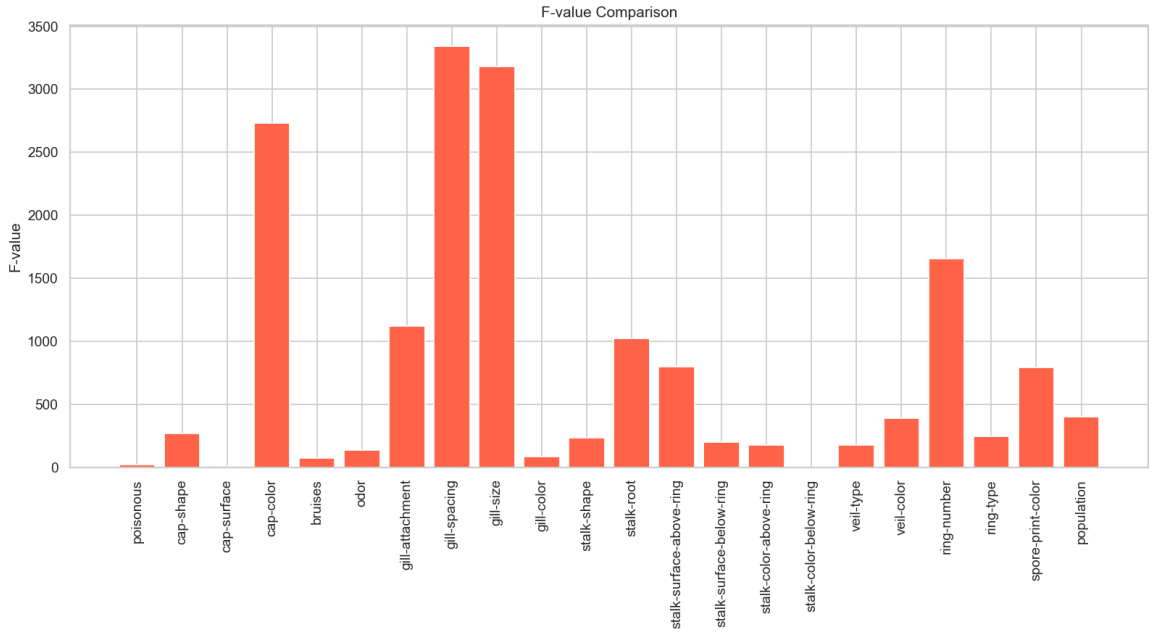


Рисунок 4.3 – Значення F-критерію ознак у навчальному наборі даних

На рис. 4.4 наведено значення дисперсії як оцінки інформативності кожної з ознак у навчальному наборі даних.

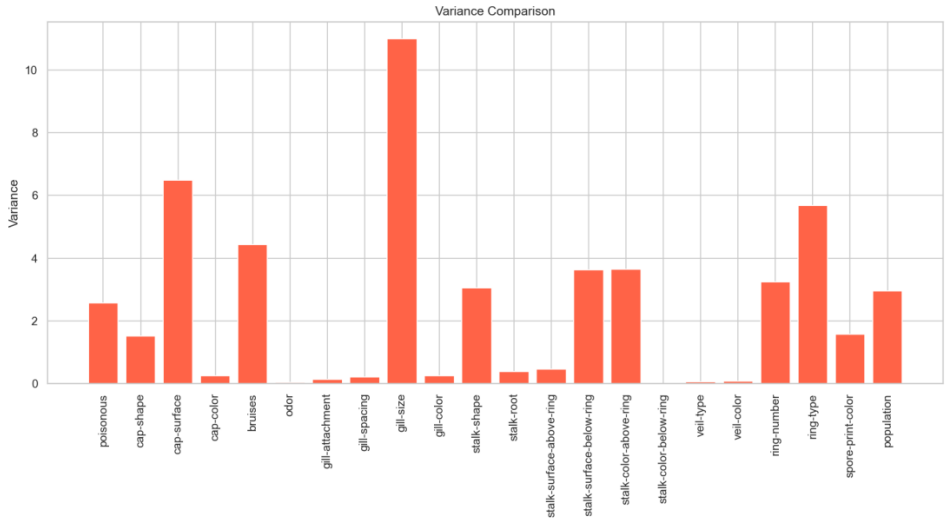


Рисунок 4.4 – Значення дисперсії ознак у навчальному наборі даних

На рис. 4.5 наведено значення критерію взаємної інформації як оцінки інформативності кожної з ознак у навчальному наборі даних.

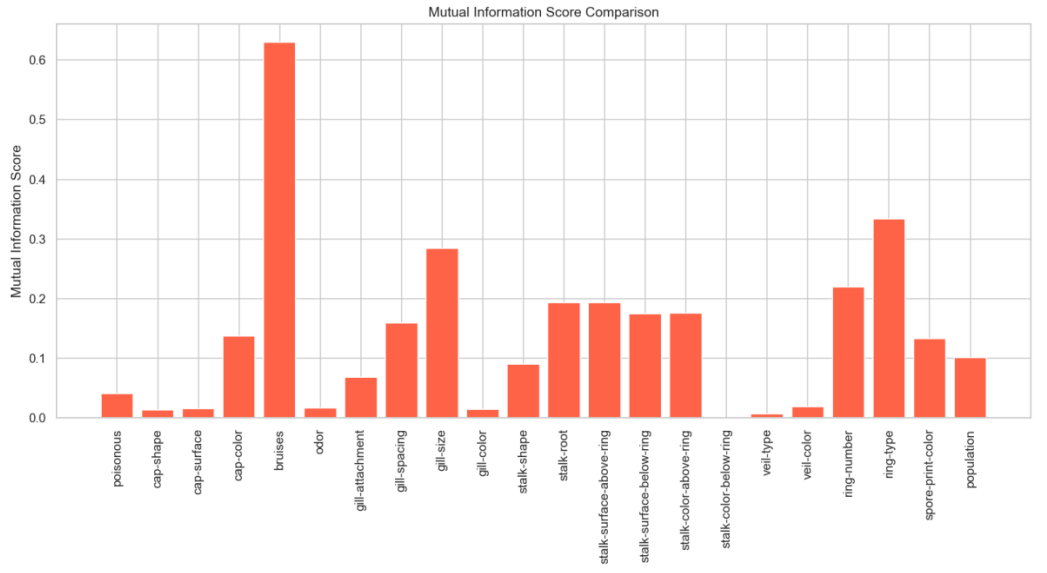


Рисунок 4.5 – Значення критерію взаємної інформації ознак у навчальному наборі даних

Графік для F-значення дав змогу виділити атрибути, що характеризуються найбільшим внеском, одним, з яких виявився параметр, пов’язаний із відстанню між пластинками гриба. Аналогічна візуалізація для показника дисперсії засвідчила, що найбільш варіативним є параметр,

пов’язаний із розміром пластинок. При аналізі взаємної інформації максимальне значення було зафіксовано для ознаки, що характеризує наявність механічних пошкоджень, тоді як діаграма для критерію хі-квадрат продемонструвала найбільший внесок з боку атрибуту, що описує розмір пластинок.

Додатково було представлено результати вимірювання часу, витраченого на виконання процедур відбору, та рівня точності моделі, отриманої після застосування різних методик (табл. 4.1).

Таблиця 4.1 – Результати експериментальних досліджень методів відбору ознак

Метод відбору ознак	Час роботи методу, сек	Точність моделі, синтезованої на основі відібраного набору ознак, %
Відбір ознак на основі F-значення	0,0075	87,3
Відбір ознак на основі порогу дисперсії	0,0064	88,1
Відбір ознак на основі критерію взаємної інформації	0,0082	90,4
Відбір з послідовним додаванням ознак	0,1045	93,2
Відбір ознак на основі генетичного алгоритму	32,154	98,7
Запропонований метод відбору ознак	3,281	98,2

Отримані результати показали динаміку продуктивності у випадку стандартних алгоритмів і при використанні розробленого методу, що дало

змогу наочно порівняти ефективність підходів та зробити висновки щодо переваг нового рішення.

4.5 Тестування програмного забезпечення, що реалізує методи відбору ознак

Виконано тестування програмного забезпечення, що реалізує методи відбору ознак.

Виявлено, що програма, що реалізує методи відбору ознак функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу вибору інформативної комбінації ознак.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов’язаних з підтримкою відбору інформативних ознак.

4.6 Висновки за розділом 4

Описано програмне забезпечення, що реалізує методи відбору ознак. Виконано тестування розробленого програмного забезпечення, що реалізує методи відбору ознак. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів вибору інформативної комбінації ознак.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процеси обчислень, пов'язані з відбором інформативних ознак.

Визначено, що відбір ознак є підходом до скорочення розмірності, що передбачає вибір обмеженої підмножини релевантних ознак з первинного набору даних шляхом усунення шумових, надлишкових або малозначущих елементів. Виконання такого відбору здатне підвищити ефективність навчання моделей, забезпечити більшу точність прогнозування, зменшити обчислювальні витрати та полегшити інтерпретацію результатів. Відзначено, що надмірність даних у сучасних великих наборах змушує застосовувати розумні алгоритми для виявлення критично важливої інформації, що забезпечує ефективне формування моделей у задачах класифікації, регресії або кластеризації. Використання методів відбору ознак дозволяє скоротити обсяг даних для обробки, зменшити складність моделей, прискорити навчання та знизити ризик перенавчання. Проаналізовано методи відбору інформативних ознак.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для аналізу даних та відбору ознак. Проте деякі програмні засоби, що реалізують методи відбору ознак, попри свою практичну цінність, мають низку обмежень, які впливають на якість та ефективність їх використання. Складність деяких методів призводить до значних обчислювальних витрат, що стає критичним при роботі з великими масивами даних. Крім того, реалізовані програмні рішення не завжди забезпечують достатню гнучкість, оскільки часто орієнтовані на використання певних підходів і можуть мати обмежені можливості налаштування для специфічних завдань. Ще однією суттєвою проблемою є ризик втрати важливої інформації, коли алгоритми відбору зменшують простір ознак надмірно агресивно, і, як наслідок, модель втрачає здатність до адекватного

відображення складних залежностей. В окремих випадках труднощі викликає також інтерпретація результатів, оскільки автоматизовані системи не завжди надають зрозуміле пояснення того, чому були відкинуті певні характеристики. Обмеженість інтеграційних можливостей із зовнішніми аналітичними програмними платформами та недостатня оптимізація під конкретні архітектури обчислювальних систем також ускладнюють практичне застосування таких рішень. Тому актуальною є розробка програмного забезпечення, що реалізує методи відбору ознак.

Сформульовано функціональні вимоги до програмного забезпечення, що реалізує методи відбору ознак.

Для реалізації програмного забезпечення, що реалізує методи відбору ознак обрано мову програмування Python, яка характеризується розвиненою екосистемою бібліотек і фреймворків, що забезпечують реалізацію алгоритмів машинного навчання, статистичного аналізу та роботи з великими масивами даних. Завдяки цьому знижується потреба у створенні алгоритмів з нуля, а розробка зосереджується на інтеграції, оптимізації та експериментуванні з різними методами. Крім того, мова Python відзначається високим рівнем зрозумілості синтаксису, що сприяє швидкому прототипуванню та спрощує процес командної роботи, коли над проєктом працюють як досвідчені програмісти, так і фахівці з даних, які не мають глибокого програмістського досвіду. Універсальність Python дозволяє поєднувати завдання з передобробки даних, візуалізації результатів та інтеграції моделей у прикладні системи в єдиному середовищі.

Для створення програмного забезпечення, що реалізує методи відбору ознак обрано середовище розробки PyCharm завдяки поєднанню інструментів, орієнтованих на ефективну роботу з мовою Python та бібліотеками для аналізу даних. PyCharm забезпечує інтелектуальне автодоповнення коду, зручну навігацію по проєкту та можливості швидкого рефакторингу, що істотно прискорює процес розробки й мінімізує кількість помилок. Важливою перевагою є інтеграція з віртуальними середовищами та системами керування

залежностями, що дозволяє легко організувати роботу з різними бібліотеками машинного навчання та інструментами для обробки даних. PyCharm підтримує зручні засоби налагодження та профілювання, що є необхідними при оптимізації алгоритмів відбору ознак, особливо коли обробляються великі масиви інформації.

Новизна роботи полягає у тому, що створено метод відбору ознак, який полягає у поступовій підготовці даних, виділенні ключових характеристик, застосуванні адаптивного алгоритму, перевірці отриманих результатів та їх інтерпретації, що дозволяє підвищити точність аналізу, зменшити вплив випадкових похибок та забезпечити більшу практичну значущість кінцевих висновків.

Практична цінність роботи полягає у тому, що розроблено програмне забезпечення, що реалізує методи відбору ознак.

Запропоновано структуру програмного забезпечення, що реалізує методи відбору ознак. Визначено, що структура програмного забезпечення, що реалізує методи відбору ознак, організована у вигляді п’яти взаємопов’язаних модулів: модуль завантаження та підготовки даних, модуль управління ознаками, модуль методів відбору, модуль оцінки якості відібраних ознак, модуль візуалізації результатів.

Описано функціонування програмного забезпечення, що реалізує методи відбору ознак. Визначено, що у процесі функціонування програми спочатку здійснюється підготовка вхідних даних, після чого активуються алгоритмічні процедури, здатні оцінювати внесок кожної змінної у загальну результативність. Оцінювання може базуватися на статистичних показниках, навчанні моделей чи пошуку комбінацій, що мінімізують надлишковість та шум, поступово простір ознак звужується, і програмне забезпечення формує підмножину характеристик, яка найкраще описує сутність досліджуваного явища.

Описано особливості реалізації програмного забезпечення, що реалізує методи відбору ознак. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням, що реалізує методи відбору ознак.

Виконано тестування розробленого програмного забезпечення, що реалізує методи відбору ознак. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів вибору інформативної комбінації ознак.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Feature Selection. Exhaustive Overview [Electronic resource]. – Access mode: https://medium.com/analytics-vidhya/feature-selection-extended-overview-b58f1d524c1c.

2. A Complete Guide to Feature Selection Methods [Electronic resource]. – Access mode: https://www.statology.org/complete-guide-feature-selection-methods/.

3. Feature Selection: Ten Effective Techniques with Examples [Electronic resource]. – Access mode: https://www.machinelearningplus.com/machine-learning/feature-selection/.

4. Feature Selection Techniques in Machine Learning [Electronic resource]. – Access mode: https://www.pickl.ai/blog/feature-selection-machine-learning/.

5. An Overview of Feature Selection [Electronic resource]. – Access mode: https://towardsdatascience.com/an-overview-of-feature-selection-1c50965551dd/.

6. Feature Selection Techniques in Machine Learning [Electronic resource]. – Access mode: https://www.stratascratch.com/blog/feature-selection-techniques-in-machine-learning/.

7. Feature Selection in Machine Learning: How to Choose the Best Features for Your Model [Electronic resource]. – Access mode: https://www.skillcamper.com/blog/feature-selection-in-machine-learning-how-to-choose-the-best-features-for-your-model.

8. Top 25 Data Mining Software [Electronic resource]. – Access mode: https://www.predictiveanalyticstoday.com/top-data-mining-software/.

9. Best Data Mining Software for Small to Big Businesses [Electronic resource]. – Access mode: https://geekflare.com/software/best-data-mining-software/.

10. Best Data Mining Tools [Electronic resource]. – Access mode: https://hevodata.com/learn/data-mining-tools/.

11. Top Data Mining Tools You Need to Know [Electronic resource]. – Access mode: https://www.simplilearn.com/data-mining-tools-article.

12. Top Data Mining Tools [Electronic resource]. – Access mode: https://www.taazaa.com/data-mining-tools/.

13. Rapid Insight Veera [Electronic resource]. – Access mode: https://eab.com/solutions/rapid-insight/.

14. Alteryx Analytics [Electronic resource]. – Access mode: https://www.alteryx.com/.

15. Teradata [Electronic resource]. – Access mode: https://www.teradata.com/.

16. Python Introduction [Electronic resource]. – Access mode: https://www.geeksforgeeks.org/python/introduction-to-python/.

17. Python [Electronic resource]. – Access mode: https://cybernetics.fandom.com/ru/wiki/Python.

18. PyCharm The only Python IDE you need [Electronic resource]. – Access mode: https://www.jetbrains.com/pycharm/.

19. Pycharm. Keymaps [Electronic resource]. – Access mode: https://www.tutorialspoint.com/pycharm/pycharm_keymaps.htm.

ДОДАТОК А

Технічне завдання

Вступ

Програмне забезпечення може використовуватися, що реалізує методи відбору ознак.

А.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація методів відбору ознак», затверджене наказом Національного університету «Запорізька політехніка» № 447 від 30 вересня 2025 р.

А.2 Призначення розробки

Програмний продукт призначений, що реалізує методи відбору інформативних ознак.

А.3 Основні вимоги до програми, що розробляється

А.3.1 Вимоги до функціональних характеристик

Програмне забезпечення, що реалізує методи відбору ознак забезпечує виконання таких функцій:

– підтримка можливості завантаження вхідного набору даних;

– підтримка можливості виконання відбору ознак за допомогою різних методів;

– можливість отримання результатів відбору ознак, зокрема, набору інформативних ознак, значення загальної інформативності набору ознак та значення індивідуальної інформативності ознак;

– можливість збереження редукованої вибірки даних;

– підтримка можливості збереження результатів відбору ознак.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення, що реалізує методи відбору ознак повинен бути зручним для користувачів.

А.3.3 Вимоги до надійності

Програмне забезпечення, що реалізує методи відбору ознак повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення, що реалізує методи відбору ознак необхідна наявність персонального комп’ютера.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення, що реалізує методи відбору ознак необхідно таке апаратне та програмне забезпечення:

– на апаратному рівні необхідно, щоб центральний процесор мав тактову частоту не нижче 4 ГГц, оскільки виконання алгоритмів відбору ознак передбачає багатократні ітерації та обчислення статистичних характеристик. Рекомендується використання багатоядерного процесора, що дозволить прискорити паралельну обробку даних. Обсяг оперативної пам’яті має бути достатнім для завантаження вибірок та проміжних результатів під час обчислень, тому мінімально необхідно чотири гігабайти, а оптимально – від восьми та вище, особливо при роботі з великими наборами даних. Для зберігання результатів експериментів та проміжних файлів рекомендується

наявність жорсткого диска або твердотільного накопичувача з обсягом пам’яті не менше п’ятидесяти гігабайтів;

– на апаратному рівні необхідна операційна система. Оптимально застосовувати операційні системи сімейства Linux або Windows останніх версій. Потрібно мати встановлене програмне забезпечення для роботи з мовою Python, а також спеціалізовані бібліотеки для машинного навчання та аналізу даних, такі як NumPy, Pandas, Scikit-learn, TensorFlow.

А.3.6 Вимоги до маркування і пакування

Програма, що реалізує методи відбору ознак може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення, що реалізує методи відбору ознак».

А.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення, що реалізує методи відбору ознак, є набір даних, що складається з множини змінних та цільової ознаки, яка визначає залежність у досліджуваній задачі. До вхідних даних також можуть належати параметри налаштувань, що задають типи методів відбору, критерії оцінювання та порогові значення, які впливають на процес формування підмножини ознак.

Вихідними даними програмного забезпечення, що реалізує методи відбору ознак, є скорочений набір змінних, що вважаються найбільш інформативними для побудови моделі, а також результати їх оцінювання за допомогою вибраних критеріїв.

ДОДАТОК Б

Фрагмент тексту програми

```
def rddt_fl(path):
    df = pd.read_csv(path)
    for col in df.columns:
        if df[col].dtype == object:
            df[col] = df[col].astype('category').cat.codes
    df.fillna(df.mean(), inplace=True)
    df = df.apply(lambda x: x if np.issubdtype(x.dtype,
np.number) else x.astype(float))
    return df

def rddt_fl_xy(df, tgt):
    x = df.drop(columns=[tgt])
    y = df[tgt]
    if y.dtype == object:
        y = y.astype('category').cat.codes
    y = y.fillna(y.mode()[0])
    return x, y

def rm_nv(x):
    x = x.loc[:, x.nunique() > 1]
    x = x.fillna(x.mean())
    x_std = x.std()
    low_var_cols = x_std[x_std < 1e-8].index
    x = x.drop(columns=low_var_cols)
    return x

def nrm_std(x):
    scaler = StandardScaler()
    x_scaled = scaler.fit_transform(x)
    df_scaled = pd.DataFrame(x_scaled, columns=x.columns)
    for col in df_scaled.columns:
        df_scaled[col] = df_scaled[col] + np.random.normal(0, 1e-
9, df_scaled[col].shape[0])
    return df_scaled

def calc_f(x, y):
    f_vals, p_vals = f_classif(x, y)
    f_series = pd.Series(f_vals, index=x.columns)
    f_series = f_series + np.random.normal(0, 1e-9,
len(f_series))
    f_series = f_series.sort_values(ascending=False)
    return f_series
```

```
def calc_chi(x, y):
    mm = MinMaxScaler()
    x_scaled = mm.fit_transform(x)
    chi_vals, p_vals = chi2(x_scaled, y)
    chi_series = pd.Series(chi_vals, index=x.columns)
    chi_series = chi_series + np.random.normal(0, 1e-9,
len(chi_series))
    chi_series = chi_series.sort_values(ascending=False)
    return chi_series

def calc_mi(x, y):
    mi_vals = mutual_info_classif(x, y, discrete_features='auto')
    mi_series = pd.Series(mi_vals, index=x.columns)
    mi_series = mi_series + np.random.normal(0, 1e-9,
len(mi_series))
    mi_series = mi_series.sort_values(ascending=False)
    return mi_series

def hyb_slt(x, y):
    x = rm_nv(x)
    f_series = calc_f(x, y)
    chi_series = calc_chi(x, y)
    mi_series = calc_mi(x, y)
    th_f = f_series.mean()
    th_chi = chi_series.mean()
    th_mi = mi_series.mean()
    slctd = []
    for clm in x.columns:
        if f_series[clm] > th_f and chi_series[clm] > th_chi and
mi_series[clm] > th_mi:
            slctd.append(clm)
        else:
            if (f_series[clm] > f_series.median() and
chi_series[clm] > chi_series.median()):
                slctd.append(clm)
    final_df = x[slctd].copy()
    for clm in final_df.columns:
        final_df[clm] = final_df[clm] + np.random.normal(0, 1e-
10, final_df.shape[0])
    return final_df

def rm_dplc(x, y):
    slctd = list(x.columns)
    corr_matrix = x.corr()
```

```
    for c1, c2 in combinations(slctd, 2):
        if corr_matrix.loc[c1, c2] > 0.85:
            mi_c1 = mutual_info_classif(x[[c1]], y)[0]
            mi_c2 = mutual_info_classif(x[[c2]], y)[0]
            if mi_c1 < mi_c2:
                if c1 in slctd:
                    slctd.remove(c1)
            else:
                if c2 in slctd:
                    slctd.remove(c2)
    final_df = x[slctd].copy()
    for clm in final_df.columns:
        final_df[clm] = final_df[clm] / (final_df[clm].max() +
1e-9)
    return final_df

def fnl_slct(path, tgt):
    df = rddt_fl(path)
    x, y = rddt_fl_xy(df, tgt)
    x_std = nrm_std(x)
    x_hyb = hyb_slt(x_std, y)
    x_final = rm_dplc(x_hyb, y)
    for col in x_final.columns:
        x_final[col] = x_final[col] + np.random.normal(0, 1e-10,
x_final[col].shape[0])
    return x_final


def scl_mnmx(x):
    mm = MinMaxScaler()
    x_scaled = mm.fit_transform(x)
    df_scaled = pd.DataFrame(x_scaled, columns=x.columns)
    for col in df_scaled.columns:
        df_scaled[col] = df_scaled[col] * np.random.uniform(0.99,
1.01, df_scaled.shape[0])
    return df_scaled

def cmp_corr(x):
    corr_mtrx = x.corr()
    high_corr_pairs = []
    for c1, c2 in combinations(x.columns, 2):
        if abs(corr_mtrx.loc[c1, c2]) > 0.75:
            high_corr_pairs.append((c1, c2, corr_mtrx.loc[c1,
c2]))
```

```
    return high_corr_pairs

def aggr_ftr(x, method="sum"):
    if method == "sum":
        df_aggr = pd.DataFrame()
        for i, col in enumerate(x.columns):
            new_col = x[col].copy()
            for j, col2 in enumerate(x.columns):
                if i != j:
                    new_col += x[col2]
            df_aggr[f"aggr_{col}"] = new_col / (len(x.columns) -
1)
    elif method == "mean":
        df_aggr = x.copy()
        df_aggr = df_aggr.apply(lambda r: r.mean(), axis=1)
    return df_aggr

def rnk_ftr(x, y):
    f_vals = calc_f(x, y)
    chi_vals = calc_chi(x, y)
    mi_vals = calc_mi(x, y)
    rank_df = pd.DataFrame()
    rank_df['f'] = f_vals.rank(ascending=False)
    rank_df['chi'] = chi_vals.rank(ascending=False)
    rank_df['mi'] = mi_vals.rank(ascending=False)
    rank_df['avg_rank'] = rank_df.mean(axis=1)
    rank_df = rank_df.sort_values('avg_rank')
    return rank_df

def sel_top(x, y, n=10):
    rank_df = rnk_ftr(x, y)
    top_features = rank_df.head(n).index.tolist()
    return x[top_features]

def rm_low_var(x, thresh=0.01):
    variances = x.var()
    keep_cols = variances[variances > thresh].index.tolist()
    df_filtered = x[keep_cols].copy()
    return df_filtered

def crs_val(x, y, folds=5):
    from sklearn.model_selection import KFold
    kf = KFold(n_splits=folds, shuffle=True, random_state=42)
    splits = []
```

```
    for train_idx, test_idx in kf.split(x):
        splits.append((x.iloc[train_idx], x.iloc[test_idx],
y.iloc[train_idx], y.iloc[test_idx]))
    return splits

def calc_pair_mi(x):
    cols = x.columns
    mi_matrix = pd.DataFrame(np.zeros((len(cols), len(cols))),
index=cols, columns=cols)
    for c1, c2 in combinations(cols, 2):
        mi_val = mutual_info_classif(x[[c1]], x[c2],
discrete_features='auto')[0]
        mi_matrix.loc[c1, c2] = mi_val
        mi_matrix.loc[c2, c1] = mi_val
    return mi_matrix

def add_noise(x, level=1e-5):
    df_noisy = x.copy()
    for col in df_noisy.columns:
        df_noisy[col] = df_noisy[col] + np.random.normal(0,
level, df_noisy.shape[0])
    return df_noisy


def smpl_ftr(x, y, pct=0.5):
    n_ftr = int(len(x.columns) * pct)
    sampled_cols = np.random.choice(x.columns, n_ftr,
replace=False)
    return x[sampled_cols]

def std_ftr(x):
    df_std = x.copy()
    for col in df_std.columns:
        df_std[col] = (df_std[col] - df_std[col].mean()) /
(df_std[col].std() + 1e-8)
    return df_std

def bnn_ftr(x, bins=10):
    df_bnn = x.copy()
    for col in df_bnn.columns:
        df_bnn[col] = pd.cut(df_bnn[col], bins=bins,
labels=False)
    return df_bnn
```

```
def ftr_encd(x):
    df_encd = x.copy()
    for col in df_encd.select_dtypes(include=['object',
'category']).columns:
        df_encd[col] = df_encd[col].astype('category').cat.codes
    return df_encd

def sel_low_corr(x, y, thresh=0.1):
    corr_with_y = x.apply(lambda c: np.corrcoef(c, y)[0, 1])
    keep_cols = corr_with_y[abs(corr_with_y) >
thresh].index.tolist()
    return x[keep_cols]

def comb_ftr(x, method="add"):
    df_comb = x.copy()
    col_list = x.columns.tolist()
    for i, c1 in enumerate(col_list):
        for j, c2 in enumerate(col_list):
            if i < j:
                if method == "add":
                    df_comb[f"{c1}_{c2}_add"] = x[c1] + x[c2]
                elif method == "mul":
                    df_comb[f"{c1}_{c2}_mul"] = x[c1] * x[c2]
                elif method == "sub":
                    df_comb[f"{c1}_{c2}_sub"] = x[c1] - x[c2]
    return df_comb

def drop_high_miss(x, thresh=0.2):
    missing_ratio = x.isnull().mean()
    keep_cols = missing_ratio[missing_ratio <=
thresh].index.tolist()
    return x[keep_cols]

def fill_miss(x, method="mean"):
    df_filled = x.copy()
    for col in df_filled.columns:
        if method == "mean":
            df_filled[col] =
df_filled[col].fillna(df_filled[col].mean())
        elif method == "median":
            df_filled[col] =
df_filled[col].fillna(df_filled[col].median())
        elif method == "mode":
```

```
            df_filled[col] =
df_filled[col].fillna(df_filled[col].mode()[0])
    return df_filled

def ftr_stb_test(x, y, n_iter=10):
    stable_score = pd.Series(index=x.columns, data=0)
    for i in range(n_iter):
        x_smpl, y_smpl = smpl_ftr(x, y, pct=0.8),
y.sample(frac=0.8)
        ranks = rnk_ftr(x_smpl, y_smpl)
        stable_score[ranks.head(10).index] += 1
    stable_score /= n_iter
    return stable_score

def exp_ftr(x, power=2):
    df_exp = x.copy()
    for col in df_exp.columns:
        df_exp[f"{col}_pow{power}"] = df_exp[col] ** power
    return df_exp


def gn_ftr_pop(x, y, pop_size=20):
    n_ftr = x.shape[1]
    pop = []
    for _ in range(pop_size):
        chrom = np.random.choice([0,1], size=n_ftr)
        pop.append(chrom)
    return np.array(pop)

def gn_ftr_fit(pop, x, y, model):
    scores = []
    for chrom in pop:
        selected_cols = x.columns[chrom.astype(bool)]
        if len(selected_cols) == 0:
            scores.append(0)
            continue
        x_sel = x[selected_cols]
        model.fit(x_sel, y)
        score = model.score(x_sel, y)
        scores.append(score)
    return np.array(scores)

def gn_ftr_sel(pop, scores, k=0.5):
    n_sel = max(1, int(len(pop)*k))
```

```
    idx = np.argsort(scores)[-n_sel:]
    return pop[idx]

def gn_ftr_cros(parents, cross_rate=0.8):
    n_chrom = parents.shape[1]
    offsprings = []
    for i in range(0, len(parents), 2):
        p1, p2 = parents[i], parents[(i+1)%len(parents)]
        if np.random.rand() < cross_rate:
            pt = np.random.randint(1, n_chrom-1)
            off = np.concatenate([p1[:pt], p2[pt:]])
            off2 = np.concatenate([p2[:pt], p1[pt:]])
            offsprings.append(off)
            offsprings.append(off2)
        else:
            offsprings.append(p1.copy())
            offsprings.append(p2.copy())
    return np.array(offsprings)

def gn_ftr_mut(pop, mut_rate=0.05):
    for chrom in pop:
        for i in range(len(chrom)):
            if np.random.rand() < mut_rate:
                chrom[i] = 1 - chrom[i]
    return pop

def gn_ftr_run(x, y, model, n_gen=20, pop_size=20,
cross_rate=0.8, mut_rate=0.05, sel_rate=0.5):
    pop = gn_ftr_pop(x, y, pop_size)
    for gen in range(n_gen):
        scores = gn_ftr_fit(pop, x, y, model)
        pop_sel = gn_ftr_sel(pop, scores, sel_rate)
        pop_off = gn_ftr_cros(pop_sel, cross_rate)
        pop = gn_ftr_mut(pop_off, mut_rate)
    final_scores = gn_ftr_fit(pop, x, y, model)
    best_idx = np.argmax(final_scores)
    best_chrom = pop[best_idx]
    best_cols = x.columns[best_chrom.astype(bool)]
    return x[best_cols], best_chrom, final_scores[best_idx]
```

ДОДАТОК В

Слайди презентації

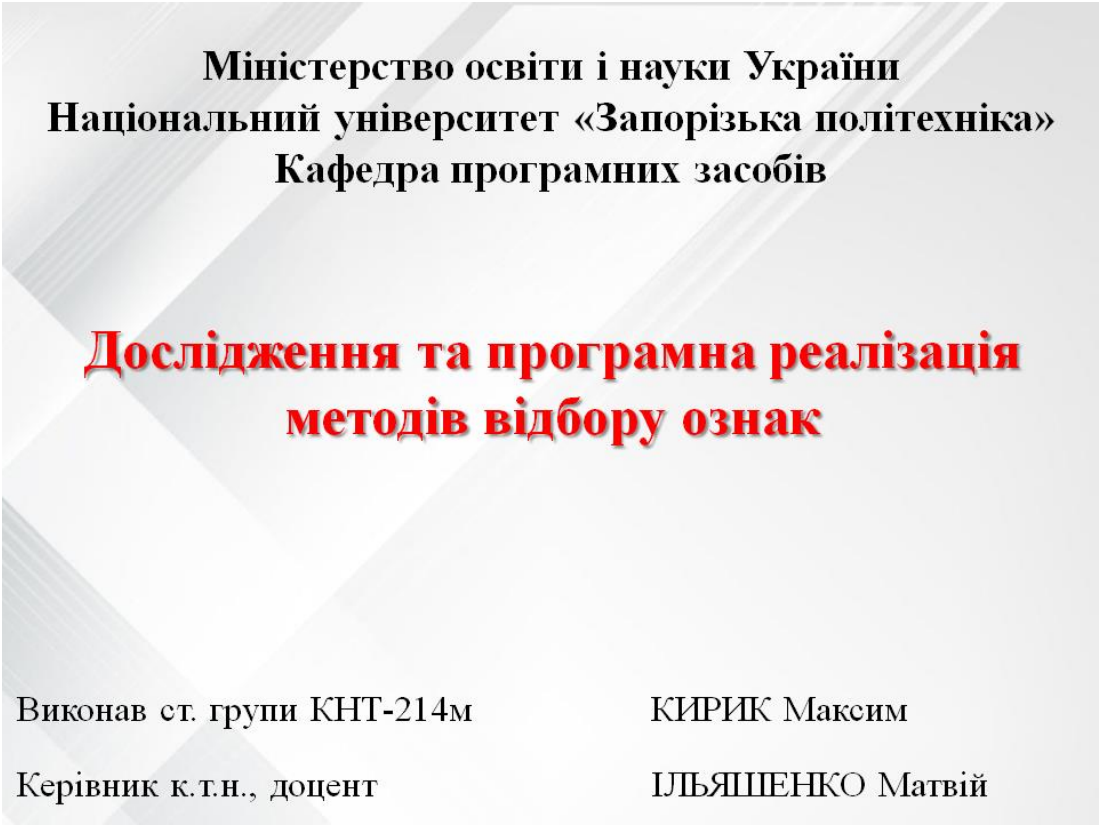


Рисунок В.1 – Слайд 1

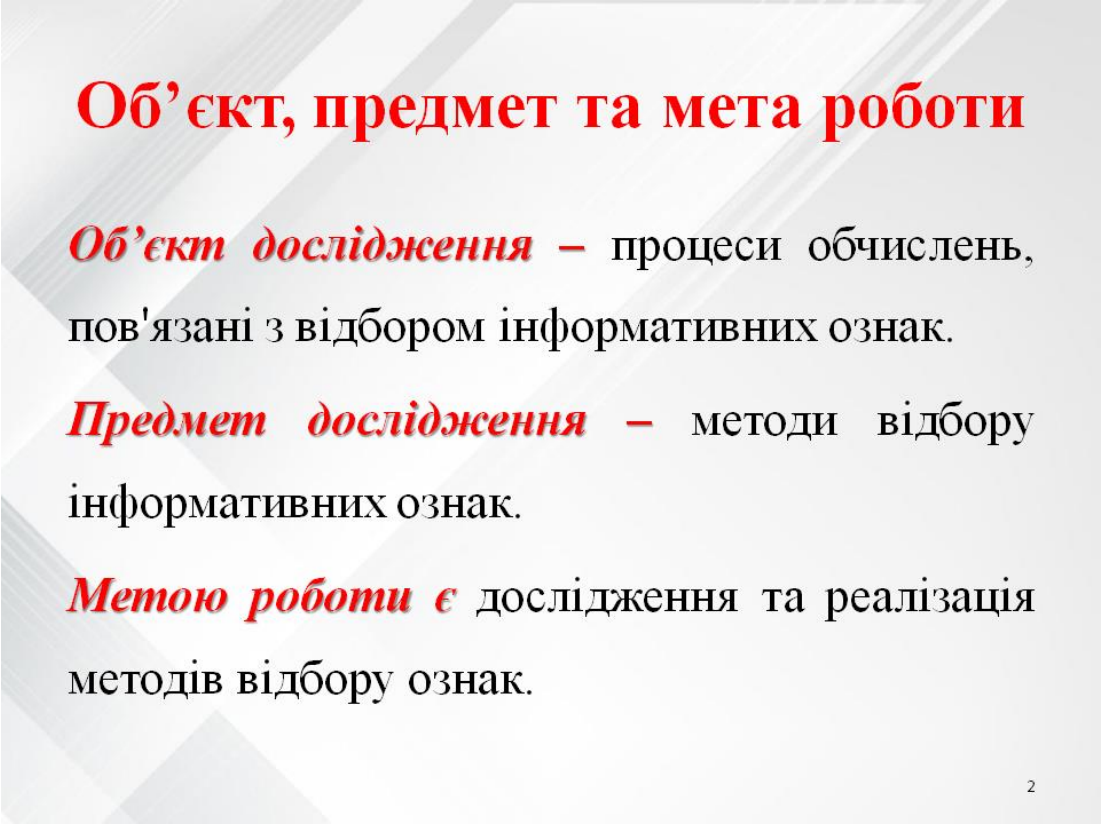


Рисунок В.2 – Слайд 2

Завдання роботи

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

– виконати методів та засобів відбору ознак;

– розробити метод відбору інформативних ознак;

– здійснити проєктування програмного забезпечення, що реалізує методи відбору ознак;

– створити програмне забезпечення, що реалізує методи відбору ознак;

– виконати тестування розробленого програмного забезпечення, що реалізує методи відбору ознак.

3

Рисунок В.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	Rapid Insight Veera	Alteryx Analytics	Teradata
Простота використання	+	+–	–
Швидкість обробки даних	+–	+	+
Масштабованість	–	+–	+
Можливості інтеграції з іншими системами	+–	+	+
Інструменти візуалізації	+–	+	+–
Аналітичні можливості	+–	+	+–

4

Рисунок В.4 – Слайд 4

Обґрунтування вибору мови програмування

Критерій порівняння мов програмування	Мова програмування		
	Python	C++	Java
Швидкість розробки прототипів	+	–	+–
Бібліотеки для машинного навчання та відбору ознак	+	+–	+–
Інтеграція з іншими інструментами	+	+–	+
Засоби візуалізації даних	+	–	+–
Популярність у науковій спільноті	+	+–	+–

5

Рисунок В.5 – Слайд 5

Вибір середовища розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	PyCharm	Jupyter Notebook	Spyder
Зручність написання та структурування проєктів ПЗ	+	–	+–
Інтелектуальне автодоповнення та рефакторинг	+	–	+
Робота з віртуальними середовищами та залежностями	+	+–	+–
Зручність для аналізу та візуалізації даних	+	+	+–
Засоби налагодження та профілювання	+	–	+–

6

Рисунок В.6 – Слайд 6

Рисунок В.7 – Слайд 7

Рисунок В.8 – Слайд 8

Рисунок В.9 – Слайд 9

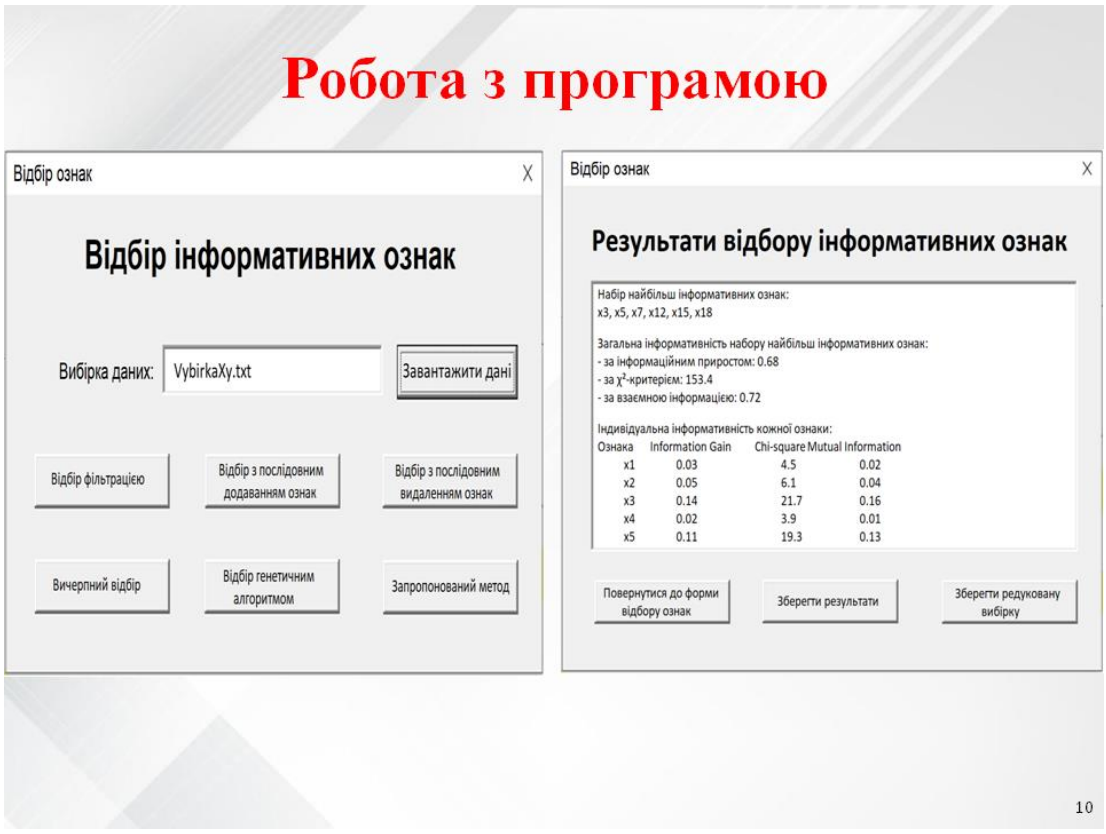
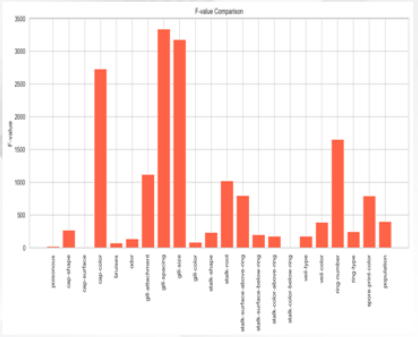
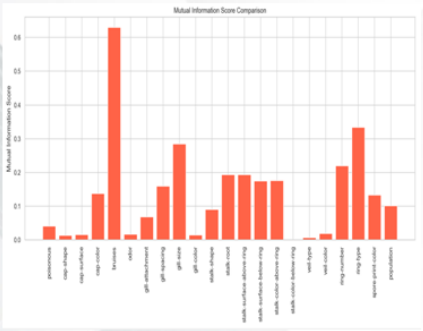


Рисунок В.10 – Слайд 10

Експерименти та результати





Метод відбору ознак	Час роботи методу, сек	Точність моделі, синтезованої на основі відібраного набору ознак, %
Відбір ознак на основі F-значення	0,0075	87,3
Відбір ознак на основі порогу дисперсії	0,0064	88,1
Відбір ознак на основі критерію взаємної інформації	0,0082	90,4
Відбір з послідовним додаванням ознак	0,1045	93,2
Відбір ознак на основі генетичного алгоритму	32,154	98,7
Запропонований метод відбору ознак	3,281	98,2

11

Рисунок В.11 – Слайд 11

Висновки

В ході виконання даної дипломної кваліфікаційної роботи було проаналізовано та досліджено процеси обчислень, пов'язані з відбором інформативних ознак.

Новизна роботи полягає у тому, що створено метод відбору ознак, який полягає у поступовій підготовці даних, виділенні ключових характеристик, застосуванні адаптивного алгоритму, перевірці отриманих результатів та їх інтерпретації.

Практична цінність результатів роботи полягає у тому, що розроблено програмне забезпечення, що реалізує методи відбору ознак.

Усі завдання випускної дипломної кваліфікаційної роботи повністю виконано.

12

Рисунок В.12 – Слайд 12