НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
КАФЕДРА ІНФОРМАТИКИ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

КУРСОВА РОБОТА

з дисципліни «Аналіз даних в інформаційних системах»

на тему: «Аналіз класифікації та прогнозування знижок на ігри PlayStation 4 та 5 за допомогою методу найближчих сусідів KNN та класифікатора дерева рішень DTC»

Студента 2 курсу групи ІП-13
Спеціальності: 121
«Інженерія програмного забезпечення»
Романюк Діани Олексіївни

«ПРИЙНЯВ» з оцінкою

_____

доц. Ліхоузова Т.А. / доц. Олійник Ю.О.

_____

Підпис Дата

Київ - 2023 рік

Національний технічний університет України “КПІ ім. Ігоря Сікорського”
Кафедра інформатики та програмної інженерії
Дисципліна Аналіз даних в інформаційно-управляючих системах
Спеціальність 121 "Інженерія програмного забезпечення"

Курс 2 Група ІП-13 Семестр 4

ЗАВДАННЯ
на курсову роботу студента
Романюк Діани Олексіївни

1.Тема роботи Аналіз класифікації та прогнозування знижок на ігри PlayStation 4 та 5 за допомогою методу найближчих сусідів KNN та класифікатора дерева рішень DTC

2.Строк здачі студентом закінченої роботи 08.06.2022

3. Вхідні дані до роботи методичні вказівки до курсової робота, обрані дані з сайту https://www.kaggle.com/datasets/shivamb/all-playstation-4-games https://www.kaggle.com/datasets/caitpj01/all-playstation-store-deals-with-ranking-eu

4.Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)
1.Постановка задачі
2.Аналіз предметної області
3.Розробка сховища даних
4.Інтелектуальний аналіз даних

5.Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

6.Дата видачі завдання 30.05.2023

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів курсової роботи	Термін виконання етапів роботи	Підписи керівника, студента
1.	Отримання теми курсової роботи	30.05.2023	
2.	Визначення зовнішніх джерел даних	30.05.2023	
3.	Пошук та вивчення літератури з питань курсової роботи	30.05.2023	
4.	Розробка моделі сховища даних	01.06.2023	
4.	Розробка ETL процесів	02.06.2023	
5.	Обґрунтування методів інтелектуального аналізу даних	03.06.2023	
6.	Застосування та порівняння ефективності методів інтелектуального аналізу даних	04.06.2023	
7.	Підготовка пояснювальної записки	06.06.2023	
8.	Здача курсової роботи на перевірку	08.06.2023	
9.	Захист курсової роботи		

Студент _____ Романюк Д. О.

(підпис) (прізвище, ім'я, по батькові)

Керівник _____ доц. Ліхоузова Т.А

(підпис) (прізвище, ім'я, по батькові)

Керівник _____ доц. Олійник Ю.О.

(підпис) (прізвище, ім'я, по батькові)

"8" червня 2023 р.

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 46 сторінок, 40 рисунків, 5 таблиць, 11 посилань.

Об’єкт дослідження: інтелектуальний аналіз даних.

Предмет дослідження: створення програмного забезпечення, що проводить аналіз даних з подальшим прогнозуванням та графічним відображенням результатів.

Мета роботи: проектування та реалізація сховища даних та ETL процесів, а також реалізація програмного забезпечення для отримання даних зі сховища та їх подальшого аналізу та прогнозування.

Дана курсова робота включає в себе: опис проектування, створення та заповнення сховища даних за даною задачею за допомогою фізичної моделі бази даних, опис створення програмного забезпечення для інтелектуального аналізу даних, їх графічного відображення та прогнозування за допомогою різних моделей.

СХОВИЩЕ ДАННИХ, МОДЕЛЬ ПРОГНОЗУВАННЯ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, ФІЗИЧНА МОДЕЛЬ БАЗИ ДАНИХ, ELT ПРОЦЕСИ

ЗМІСТ

ВСТУП

Знижки на ігри для платформ PlayStation 4 та 5 завжди привертають увагу геймерів та шукачів вигідних пропозицій. Спеціальні акції та знижки роблять широкий асортимент ігор більш доступними. При плануванні покупки, гравці часто стикаються з питаннями: які ігри варто купувати, які знижки є найвигіднішими, чи є можливість передбачити, наскільки великою буде знижка на в майбутньому? Відповіді на ці питання можуть допомогти прийняти обґрунтоване рішення та зекономити кошти.

У даній роботі ми проведемо аналіз класифікації та прогнозування знижок на ігри для PlayStation 4 та 5 з використанням методу найближчих сусідів (KNN) та класифікатора дерева рішень (DTC). Ми будемо використовувати набір даних, що містить інформацію про оригінальну ціну, знижку, рік випуску та інші параметри ігор.

Основна мета роботи полягатиме у розробці моделей, які зможуть класифікувати знижки на ігри в залежності від їх параметрів та прогнозувати величину знижки на основі історичних даних. Це дозволить нам зрозуміти, які фактори впливають на розмір знижки та які ігри можуть мати найбільші шанси на отримання значної знижки. Отримані моделі можуть бути використані розробниками та видавництвами ігор для стратегічного планування цін та знижок на свої продукти.

Таким чином, аналіз класифікації та прогнозування знижок на ігри для PlayStation 4 та 5 має великий потенціал у полі планування покупок ігор та розвитку галузі відеоігор в цілому. Далі ми детально розглянемо методику проведення дослідження та отримані результати, які будуть цінними для широкого спектру зацікавлених осіб.

1.ПОСТАНОВКА ЗАДАЧІ

Під час виконання курсової роботи необхідно виконати наступні завдання: Створення сховища даних типу «сніжинка». Структура курсової роботи узгоджена з керівником. Створення ETL процесів для завантаження даних до сховища, їх отримання зі самого сховища за допомогою запитів. Реалізувати спроектоване сховище даних з використанням PostgreSQL версії 15.2. Після обробки даних створити новий датасет, що відповідає темі роботи на основі трьох попередніх файлів. Також виконання даної курсової роботи потребує виконання декількох задач: аналізу предметної області; роботи з датасетом: завантаження, дослідження його структури та виправлення наявних помилок; вибір методів для прогнозування та обґрунтування даного вибору; аналіз отриманих результатів кожного з методів та порівняння отриманих результатів ефективності. Створення застосунку, що поділяє отримані дані на тренувальні та тестові для перевірки декількох методів, у подальшому порівняння ефективності методів. Для прогнозування буде використано методи K-Nearest Neighbors та Decision Tree Classifier. Для кожного методу проаналізувати результати та в кінці порівняти результати цих двох методів. Обрати найоптимальніший метод для прогнозування знижки. Виконане дослідження можна буде використовувати для прогнозування наступного зниження ціни. Вхідними даними буде набір даних, що містить інформацію про оригінальну ціну, знижку, рік випуску та інші параметри ігор.

2.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Комп'ютерні ігри вже давно перетворилися на якісну та популярну форму інтерактивної розваги, що займає вагому позицію у масовій культурі. Вони не тільки пропонують розвагу, але й стають провідниками творчості та інновацій. Для інтернет-магазинів, що продають комп'ютерні ігри, важливо розуміти закономірності успішності гри та прогнозувати її популярність. Це допоможе їм робити обґрунтовані вибори і планувати ефективні рекламні кампанії. Розробка якісних ігор та використання відповідних інструментів мають вирішальне значення для успіху на цьому ринку. Аналіз класифікації та прогнозування знижок на ігри PlayStation 4 та 5 надасть цінну інформацію для гравців та розробників, допомагаючи зробити обґрунтовані рішення та визначити потенційно популярні ігри.

У програмному забезпеченні буде реалізовано наступну функціональність, що включає в себе:

- проектування сховища даних;
- створення ETL процесів для завантаження і оновлення даних;
- створення вибірки даних з сховища;
- інтелектуальний аналіз даних;
- використання декількох моделей прогнозування даних;
- прогнозування наступних знижок на ігри;
- прогнозування факторів впливу на потенційне зниження ціни гри;
- графічне відображення отриманих результатів та їх аналіз.

3.РОЗРОБКА СХОВИЩА ДАНИХ

3.1 Розробка моделі сховища даних

Для виконання курсової роботи було обрано3 джерела відкритих даних на сайті https://www.kaggle.com//, що включають в себе:

– Статистика випуску ігор на Playstation4 та 5: https://www.kaggle.com/datasets/shivamb/all-playstation-4-games

– Статистика знижок на ігри Playstation4 та 5 (список 1): https://www.kaggle.com/datasets/caitpj01/all-playstation-store-deals-with-ranking-eu

– Статистика знижок на ігри Playstation4 та 5 (список 2): https://www.kaggle.com/datasets/caitpj01/all-playstation-store-deals-with-ranking-eu?select=Outputfirst_PSSEO_

– На основі детального опису та проведеного аналізу предметної області було розроблено наступну модель сховища даних за типом „сніжинка” (рисунок 3.1).

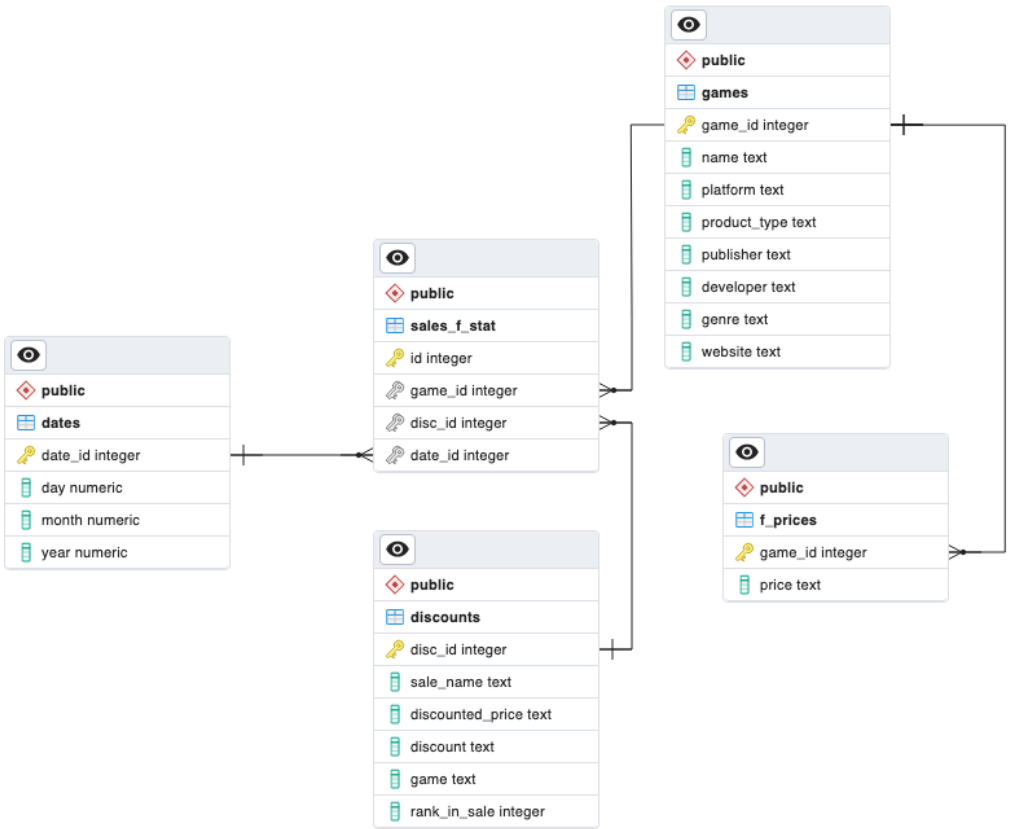


Рисунок 3.1 - Проектування моделі сховища за типом сніжинка

У моделі сховища за типом сніжинка спроектовано дві таблиці фактів та три таблиці вимірів. На даній моделі видно, що центральною таблицею вимірів є таблиця sales_f_stat, яка відповідає за продажі ігор. Нижче наведено опис полів кожної таблиці сховища даних відеоігор (таблиці 3.1-3.1.5).

Таблиця 3.1 – Таблиця виміру інформації про ігри games

Назва поля	Опис поля
1.game_id	Унікальний ідентифікатор запису гри
2.name	Назва гри
3.platform	Платформа на якій можна запустити гру
4.product_type	Тип гри
5.publisher	Видавництво гри
6.developer	Розробник гри
7.genre	Жанр гри
8.website	Сайт на якому можна запустити гру

Таблиця 3.1.2 – Таблиця виміру інформації про ігри discounts

Назва поля	Опис поля
1.disc_id	Унікальний ідентифікатор запису знижки
2.sale_name	Назва знижки
3.discounted_price	Ціна гри після знижки
4.discount	Розмір знижки
5.game	Гра на яку діє знижка

6.rank_in_sale	Місце в списку знижок

Таблиця 3.1.3 – Таблиця виміру інформації про ігри dates

Назва поля	Опис поля
1.date_id	Унікальний ідентифікатор запису дати
2.day	День
3.month	Місяць
4.year	Рік

Таблиця 3.1.4 – Фактова таблиця інформації про ігри f_prices

Назва поля	Опис поля
1.game_id	Унікальний ідентифікатор запису гри
2.price	Ціна гри

Таблиця 3.1.5 – Фактова таблиця інформації про ігри sales_f_stat

Назва поля	Опис поля
1.id	Унікальний ідентифікатор запису
2.game_id	Унікальний ідентифікатор запису гри
3. disc_id	Унікальний ідентифікатор запису знижки
4.date_id	Унікальний ідентифікатор запису дати

Створення таблиці виміру статистики ігор games в сховищі даних:

```
create table games
(
    game_id serial primary key ,
    name  text,
    platform text,
    product_type text,
    publisher   text,
    developer text,
    genre text,
    website text
);
```

Створення таблиці виміру знижок discounts в сховищі даних:

```
create table discounts
(
    disc_id serial primary key ,
    sale_name        text,
    discounted_price text,
    discount text,
    game text,
    rank_in_sale integer
);
```

Створення тимчасової таблиці дат випуску ігор в сховищі даних:

```
create table tdates
(
    date_id serial primary key ,
    day numeric,
    month numeric,
    year numeric
```

```
);
```

Створення таблиці виміру дат випуску ігор в сховищі даних:

```
create table dates
(
    date_id serial primary key ,
    day numeric,
    month numeric,
    year numeric
);
```

Створення таблиці фактів оригінальних цін на ігри f_prices в сховищі даних:

```
create table f_prices
(
    game_id serial primary key references games,
    price   text
);
```

Створення таблиці фактів продажів sales_f_stat в сховищі даних:

```
create table sales_f_stat(
    id serial primary key ,
    game_id integer references games,
    disc_id integer references discounts,
    date_id integer references dates
)
```

Створення таблиці нового датасету з використанням вище наведених даних

```
create table new_dataset
(
    id serial primary key ,
    name            text,
```

```
    platforms        text,
    original_price   text,
    discounted_price text,
    discount         text,
    product_type     text,
    publisher text,
    rel_date  date,
    developer text,
    genre     text
);
```

3.2 Розробка ETL процесів

Процес завантаження даних в ETL передбачає збір даних різних якостей з різних джерел і форматів. Вхідні дані можуть бути зібрані з різних додатків та мати різне кодування. Для ефективного аналізу ці дані повинні бути перетворені в єдиний універсальний формат, щоб уникнути складнощів, пов'язаних з різнорідним вмістом.

На етапі валідації даних проводиться перевірка на коректність і повноту. Звіт про помилки складається для подальшого виправлення та узагальнення даних. Очищення даних в процесі ETL має технічний характер і підготовлює дані для завантаження в сховище. В аналітичній системі проводиться додаткове очищення для вирішення конкретних аналітичних задач.

Перетворення даних є важливим етапом процесу ETL. Вони підготовляються для зберігання в сховищі та оптимального аналізу. Під час перетворення використовуються різноманітні інструменти, включаючи ручне редагування та складні методи обробки даних, залежно від вимог до якості даних.

Завантаження даних в сховище є останнім етапом ETL. Дані переносяться з проміжних таблиць до структур сховища даних. Для ефективного завантаження необхідно забезпечити повноту та коректність даних перед процесом завантаження. Завантаження включає перенесення тільки змінених даних, що дозволяє прискорити процес заповнення сховища.

В цілому, процес ETL забезпечує збір, очищення, перетворення та завантаження даних в сховище для подальшого аналізу. Це необхідний крок для забезпечення якості даних і раціонального використання їх в аналітичних задачах.

3.3 Завантаження даних за допомогою ETL процесів

Для заповнення сховища даних відеоігор було реалізовано наступне:

а) Завантажуємо дані з датасетів в тимчасові таблиці:

1)playstat

2)playstat1_2

3)playstat2

б) Парсимо дату виходу гри в тимчасову таблицю tdates, видаляємо повторення

в) Для завантаження даних в новий датасет чистимо дані в original_price, discounted_price та discount, видаляємо знаки “%”, “£” та “.” Для подальшого перетворення в числовий формат

Код даних процесів наведено в додатку А

Код завантаження даних до таблиць наведено в додатку А

1) Таблиця виміру games (рисунок 3.3.1).

	game_id	name	platform	product_type	publisher
1	1	10 Second Ninja X	['PS4']	GAME_BUNDLE	Curve Games
2	2	13 Sentinels: Aegis Rim	['PS4']	FULL_GAME	Atlus
3	3	20XX	['PS4']	FULL_GAME	Fire Hose Games
4	4	36 Fragments of Midnight	['PS4']	FULL_GAME	Ratalaika Games
5	5	39 Days to Mars	['PS4']	FULL_GAME	It's Anecdotal
6	6	99Vidas	['PS4']	FULL_GAME	QUByte Interactive
7	7	A Day Without Me	['PS4']	FULL_GAME	ChiliDog Interactive
8	8	A Knight's Quest	['PS4']	FULL_GAME	Curve Games
9	9	ACT IT OUT! A Game of Charades	['PS4']	FULL_GAME	Snap Finger Click
10	10	AO Tennis 2	['PS4']	FULL_GAME	Nacon, Oizumi Amuzio
11	11	ARK Park	['PS4']	FULL_GAME	Snail Games
12	12	ARK: Survival Evolved	['PS4']	FULL_GAME	Wildcard Properties
13	13	AWAY: Journey to the Unexpected	['PS4']	FULL_GAME	PLAYDIUS

Рисунок 3.3.1 - Завантаження даних у таблицю вимірів games

2) Таблиця виміру discounts(рисунок 3.3.2).

	disc_id	sale_name	discounted_price	discount	game	rank_i
62	62	WM_EU_DEALS	£4.99	-80%	Tennis World Tour - Roland-Garros Edition	
63	63	WM_EU_DEALS	£17.49	-30%	Actraiser Renaissance	
64	64	WM_EU_DEALS	£13.19	-60%	Control: Ultimate Edition	
65	65	WM_EU_DEALS	£6.99	-50%	Mars Alive	
66	66	WM_EU_DEALS	£59.19	-20%	Black Desert - 11,500 Pearls	
67	67	WM_EU_DEALS	£3.99	-50%	Corridor Z - Disturbed Dynamic Theme Bundle	
68	68	WM_EU_DEALS	£5.99	-70%	Metamorphosis	
69	69	WM_EU_DEALS	£2.87	-75%	Obliteracers	
70	70	WM_EU_DEALS	£1.99	-80%	Valthirian Arc: Hero School Story	
71	71	WM_EU_DEALS	£3.24	-75%	Strikers Edge	
72	72	WM_EU_DEALS	£13.99	-30%	Twelve Minutes	
73	73	WM_EU_DEALS	£3.14	-65%	Fishing Sim World®: Pro Tour - Jezioro Bestii	
74	74	WM_EU_DEALS	£4.99	-75%	The Metronomicon: Slay the Dance Floor - Del…	

Рисунок 3.3.2 - Завантаження даних у таблицю вимірів discounts

3) Таблиця виміру dates (рисунок 3.3.3).

	date_id	day	month	year
39	39	4	2	2020
40	40	22	1	2020
41	41	27	4	2017
42	42	3	6	2014
43	43	12	6	2020
44	44	11	12	2018

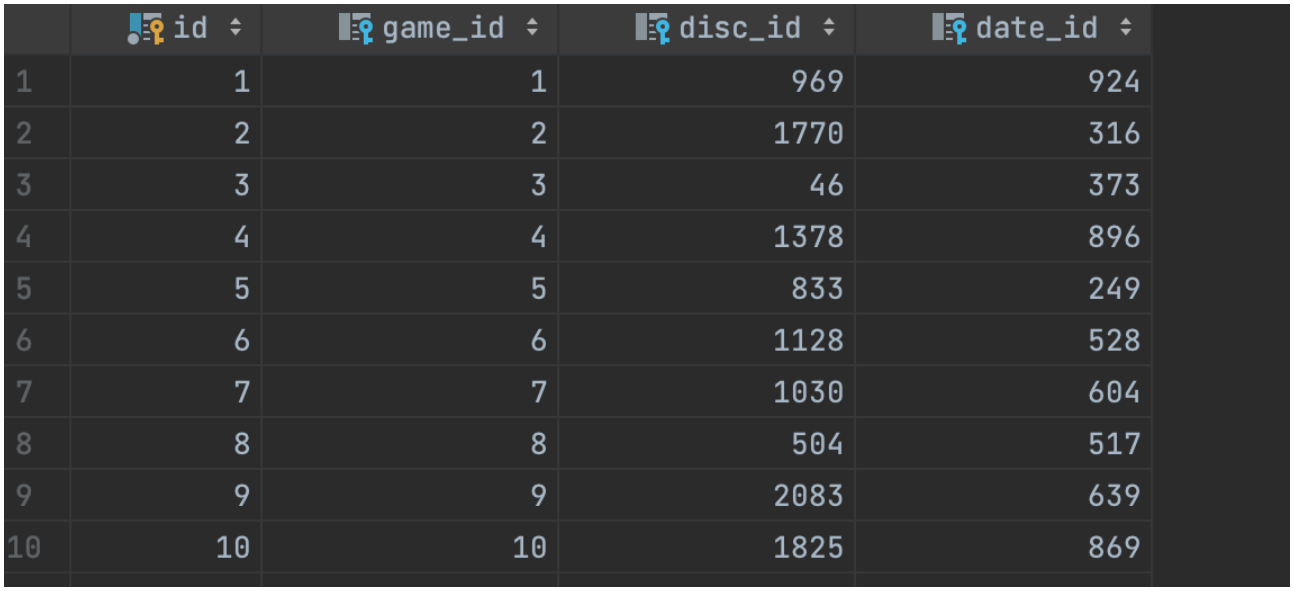
Рисунок 3.3.3 - Завантаження даних у таблицю вимірів dates

4) Таблиця фактів f_prices (рисунок 3.3.4).

	game_id	price
1	1	£7.99
2	2	£49.99
3	3	£12.99
4	4	£2.49
5	5	£9.79

Рисунок 3.3.4 - Завантаження даних у таблицю вимірів f_prices

5) Таблиця фактів sales_f_stat (рисунок 3.3.5).



	id	game_id	disc_id	date_id
1	1	1	969	924
2	2	2	1770	316
3	3	3	46	373
4	4	4	1378	896
5	5	5	833	249
6	6	6	1128	528
7	7	7	1030	604
8	8	8	504	517
9	9	9	2083	639
10	10	10	1825	869

Рисунок 3.3.5 - Завантаження даних у таблицю вимірів sales_f_stat

4.РОБОТА З ДАНИМИ

4.1 Опис обраних даних

Для вирішення поставленної перед нами задачі використаємо новий створений попередньо датасет. Даний набір даних складається з аналізів 687 ігор виданими на PlayStation4 та 5. Даний датасет містить в собі одну таблицю, що складається з 10 стовпців, а саме: name , platforms, original_price(£), discounted_price(£), discount(%), product_type, publisher, rel_date, developer, genre

Дані стовпці несуть в собі наступну інформацію:

- name – назва гри;

- platforms – платформа для якої передбачена гра;

- original_price(£) – оригінальна ціна;

- discounted_price(£) – ціна після знижки;

- discount(%) – знижка;

- product_type – тип гри;

- publisher – видавництво;

- rel_date – дата випуску гри;

- developer – компанія розробник;

- genre – жанр;

4.2 Перевірка даних

Для роботи з даними на мові Python ми використовуємо бібліотеку «pandas».

Для початку ми зчитуємо дані з файлу та виводимо основну інформацію про наш датафрейм (рис. 4.2.1), (рис 4.2.2), (рис 4.2.3)

```python
pd.set_option('display.max_columns', None)
dataframe = pd.read_csv('data.csv', delimiter=',', decimal=',', encoding='UTF-8')
dataframe.info()
print(dataframe.head(5))
```

рис. 4.2.1 Завантаження даних

```
Data columns (total 11 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   id                   688 non-null    int64
 1   name                 688 non-null    object
 2   platforms            688 non-null    object
 3   original_price(£)    688 non-null    object
 4   discounted_price(£)  687 non-null    object
 5   discount(%)          688 non-null    int64
 6   product_type         688 non-null    object
 7   publisher            688 non-null    object
 8   rel_date             688 non-null    object
 9   developer            688 non-null    object
 10  genre                688 non-null    object
```

рис. 4.2.2 Виводимо інформацію про датафрейм

```
    id                name platforms original_price(£) discounted_price(£)  \
0  414               Chess   ['PS4']              7.39                1.84
1  530             Spencer   ['PS4']              5.79                2.31
2    1        Table Tennis   ['PS4']              7.99                1.99
3    2           Rainswept   ['PS4']              7.99                4.39
4    3  Hidden Through Time   ['PS4']              6.49                3.24

   discount(%) product_type        publisher    rel_date  \
0          -75    FULL_GAME            Sabec  2020-07-30
1          -60    FULL_GAME      EntwicklerX  2021-04-09
2          -75    FULL_GAME            Sabec  2020-08-27
3          -45    FULL_GAME  2Awesome Studio  2020-07-21
4          -50    FULL_GAME        Rogueside  2020-03-12

              developer                 genre
0                 Sabec          Card & Board
1           EntwicklerX            Platformer
2                 Sabec  Sports, Table Tennis
3  Frostwood Interactive             Adventure
```

рис. 4.2.3 Виводимо перші 5 рядків

Бачимо, що колонка discount(%) набуває від'ємних значень. Замінимо його модулем. Також необхідно видалитии рядки в яких є порожні значення. Змінимо тип даних discount(%), original_price(£), discounted_price(£) на float (рис. 4.2.4)

```
dataframe.dropna(inplace=True)  # Видалення рядків з нульовими значеннями

dataframe['discounted_price(£)'] = dataframe['discounted_price(£)'].astype(float)
dataframe['original_price(£)'] = dataframe['original_price(£)'].astype(float)
dataframe['discount(%)'] = dataframe['discount(%)'].astype(float)
dataframe['discount(%)'] = dataframe['discount(%)'].abs()
```

Рис 4.2.4 Зміна типу даних, видалення нулів, зміна від'ємних значень

Зробимо перевірку (рис. 4.2.5)

```
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   id                  687 non-null    int64
 1   name                687 non-null    object
 2   platforms           687 non-null    object
 3   original_price(£)   687 non-null    float64
 4   discounted_price(£) 687 non-null    float64
 5   discount(%)         687 non-null    float64
 6   product_type        687 non-null    object
 7   publisher           687 non-null    object
 8   rel_date            687 non-null    object
 9   developer           687 non-null    object
 10  genre               687 non-null    object
dtypes: float64(3), int64(1), object(7)
```

Рис 4.2.5 Оновлена інформація про датафрейм

Кількість непорожніх значень (Non-Null Count) для всіх колонок дорівнює 687. Це означає, що всі рядки мають дані для всіх 11 колонок. Типи даних (Dtype)

для кожної колонки відповідають відповідним типам даних: int64 для колонки "id", float64 для колонок "original_price(£)", "discounted_price(£)", і "discount(%)", та object для решти колонок, які містять текстові значення. Колонка "id" містить унікальні ідентифікатори для кожного рядка, що вказує на його унікальність. Загально, датафрейм містить 687 рядків та 11 колонок і не має відсутніх значень у жодній колонці. Це свідчить про те, що датафрейм є повним і коректним для подальшого аналізу даних.

Створимо додаткову колонку, в які знижки будуть умовно поділені на 4 групи (рис.4.2.6), (рис 4.2.7):
група x:0 – незначні знижки менше 30%
група x:1 – знижки менше більше 30% та менше 50%
група x:2 – знижки що дорівнюють половині ціни
група x:3 – великі знижки більше 50%

```
dataframe['discount_label'] = dataframe['discount(%)'].apply(lambda x: 0 if x < 30 else (1 if x < 50 else (2 if x == 50 else 3)))
dataframe.info()
```

Рис 4.2.6 Додавання нового стовпчику

```
---  ------               --------------  -----
 0   id                   687 non-null    int64
 1   name                 687 non-null    object
 2   platforms            687 non-null    object
 3   original_price(£)    687 non-null    float64
 4   discounted_price(£)  687 non-null    float64
 5   discount(%)          687 non-null    float64
 6   product_type         687 non-null    object
 7   publisher            687 non-null    object
 8   rel_date             687 non-null    datetime64[ns]
 9   developer            687 non-null    object
 10  genre                687 non-null    object
 11  year                 687 non-null    float64
 12  discount_label       687 non-null    int64
```

Рис 4.2.7 Перевірка чи з’явився новий стовпчик

Створюємо допоміжний датафрейм для відокремлення даних на основі яких буде здійснюватись аналіз(рис 4.2.8)

```
data2 = dataframe.loc[:, ['discount(%)', 'year', 'original_price(£)', 'discounted_price(£)', 'discount_label']]
#-----------------------------------------------------
```

Рис 4.2.8 Допоміжний датафрейм

Перевіримо розподіл групованих знижок на ігри в залежності від їх величини (рис 4.2.9),(4.2.10)

```
sns.set(style='darkgrid')
plt.figure(figsize=(12, 6))

colors = sns.color_palette("Blues", len(data2['discount_label'].unique()))

sns.countplot(x='discount_label', data=data2, palette=colors)
plt.xlabel('Discount_label')
plt.ylabel('Count')
plt.title('Distribution of Discounts')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

Рис 4.2.9 Побудова графіку розподілу групованих знижок

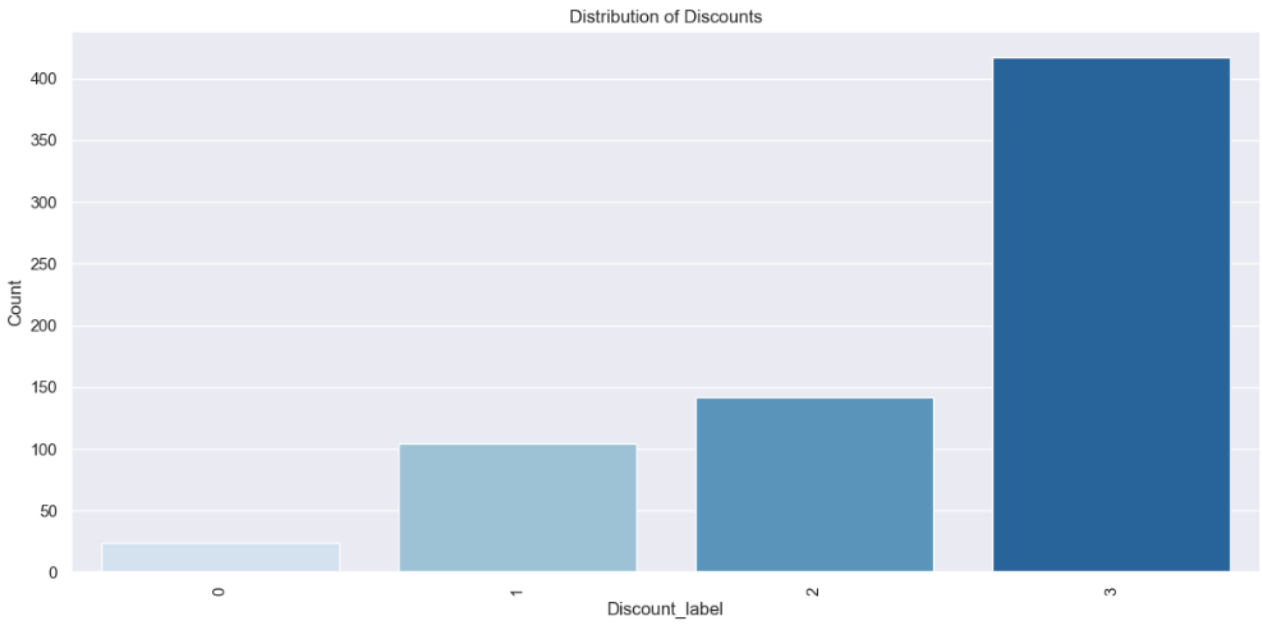


Рис 4.2.10 Відображення розподілу групованих знижок

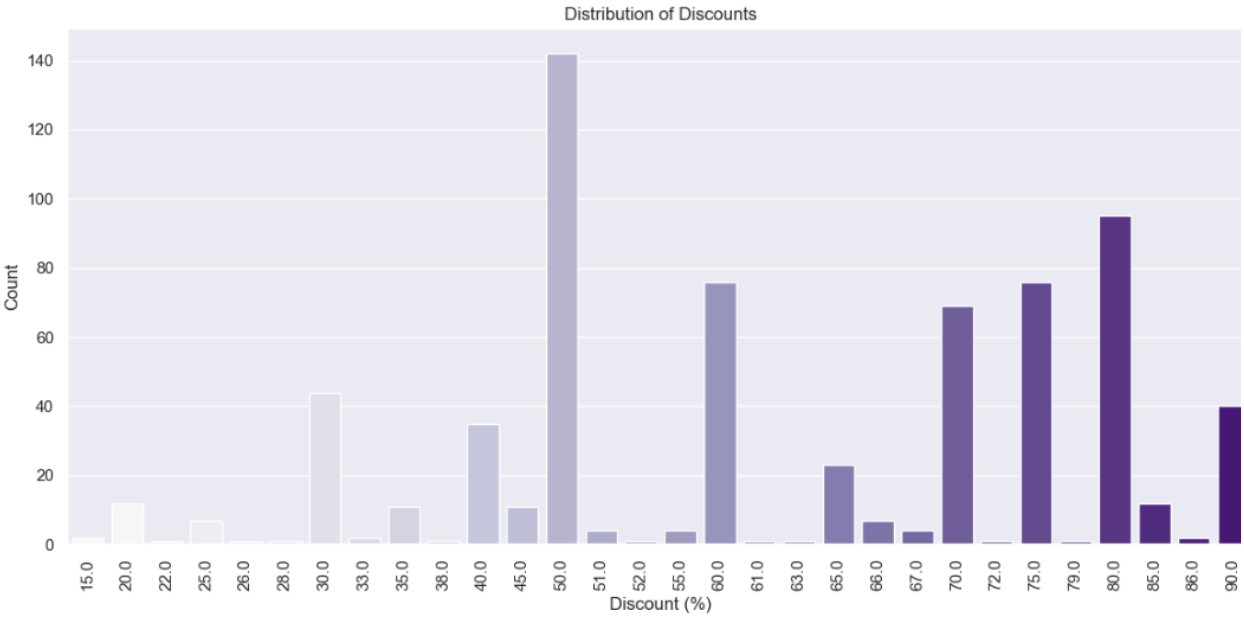


Рис 4.2.11 Відображення розподілу усіх знижок

Можемо зробити висновки, що найбільша кількість знижок 3 групи, що є більше 50%

4.3 Поділ даних

Останнім кроком ми ділимо дані на тренувальні та тестові для подальшої роботи з методами класифікації

4.3.1 Поділ для групованих знижок (рис. 4.3.1)

```
# Розділення даних на ознаки (X) та цільову змінну (y)
X = data2.drop(columns='discount_label')
y = data2['discount_label']

# Розділення на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8, random_state=0)
```

Рис 4.3.1 Поділ даних використовуємо discount_label

4.3.2 Поділ для усіх знижок (рис 4.3.2)

```
X = data2.drop(columns='discount(%)')
y = data2['discount(%)']

# Розділення на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8, random_state=0)
```

Рис 4.3.2 Поділ даних використовуємо discount(%)

Розділення даних на тренувальні і тестові вибірки виконується для оцінки та перевірки ефективності моделей машинного навчання. Головна причина розділення даних полягає в тому, що ми хочемо оцінити, наскільки добре модель вміє узагальнювати та передбачати значення на нових, раніше не бачених даних. Тренувальна вибірка використовується для навчання моделі на відомих даних, тоді як тестова вибірка використовується для оцінки ефективності моделі на невідомих даних. Процес розділення даних зазвичай включає в себе випадкове розподілення даних на тренувальну і тестову вибірки з визначеним співвідношенням. У нашому коді, train_test_split функція використовується для розділення даних на тренувальну та тестову вибірки з співвідношенням 80% до 20% відповідно. Після розділення даних, модель машинного навчання, у цьому випадку KNN-класифікатор, навчається на тренувальній вибірці за допомогою визначених параметрів. Потім використовується тестова вибірка для оцінки точності моделі на невідомих даних.

5.ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ

5.1 Обґрунтування вибору методів інтелектуального аналізу даних

Вибір методів інтелектуального аналізу даних залежить від конкретних вимог та цілей проекту, а також властивостей самого набору даних. У даному випадку, ви використовуєте методи класифікації для аналізу даних.

1) K-Nearest Neighbors (KNN) - цей метод заснований на припущенні, що подібні об'єкти знаходяться близько один до одного в просторі ознак. Він класифікує нові об'єкти, порівнюючи їх з найближчими сусідами в тренувальній вибірці. Ви використовуєте KNN для прогнозування рівнів знижок у першому і другому аналізах.

2) Decision Tree Classifier (DTC) - цей метод використовує дерево прийняття рішень для класифікації об'єктів. Воно розбиває набір даних на різні вузли за допомогою різних ознак і умов. Дерева прийняття рішень можуть бути легкими для інтерпретації і розуміння. Ви використовуєте DTC для класифікації рівнів знижок у першому аналізі.

Обрані методи інтелектуального аналізу даних відповідають задачам класифікації у наших аналізах. Кожен метод має свої переваги та обмеження, і вибір конкретного методу залежить від наших вимог та властивостей даних. У даному випадку, ви можете порівняти ефективність та точність моделей KNN і DTC на тренувальних і тестових вибірках для визначення найкращого підходу до вашої проблеми класифікації знижок.

5.2 Аналіз отриманих результатів для методу K-Nearest Neighbors

Для аналізу результатів роботи методу KNN спочатку потрібно вибрати найкращу модель з усіх можливих варіантів. Для цього ми використали модуль, який перебирає всі можливі комбінації параметрів і допомагає визначити, які параметри найкраще вирішують поставлену задачу (рис. 5.2.1).

```
# Ініціалізація KNN-класифікатора та параметрів для пошуку
classifier = KNeighborsClassifier()
parameters = {'n_neighbors': range(1, 25)}
grid_search = GridSearchCV(classifier, parameters, cv=10, verbose=1)
grid_search.fit(X_train, y_train)
print("Найкращі параметри: ", grid_search.best_estimator_)
```

Рис 5.2.1 Пошук найкращих параметрів

Після цього ми навчили модель на тренувальних даних і перевірили її точність прогнозування (рис 5.2.2).

```
# Використання найкращих параметрів для навчання моделі
best_classifier = grid_search.best_estimator_
best_classifier.fit(X_train, y_train)

# Прогнозування на тренувальній та тестовій вибірці
train_accuracy = round(best_classifier.score(X_train, y_train), 5)
test_accuracy = round(best_classifier.score(X_test, y_test), 5)

print("Точність KNN на тренувальній вибірці:", train_accuracy)
print("Точність KNN на тестовій вибірці:", test_accuracy)
```

Рис 5.2.2 Результати точності для KNN

Також перевіримо прогноз для яких інтервалів параметрів виявився не правильним побудувавши графік (рис. 5.2.3):

```
plt.rcParams["figure.figsize"] = (15,4)
plt.gca().axes.get_yaxis().set_visible(False)
plt.plot(X_test.index, y_test, "yx", label = "True result")
plt.plot(X_test.index, best_classifier.predict(X_test), "b+", label = "Predict result")
plt.legend(loc='center right', shadow=True)
plt.show()
```

Рис 5.2.3 Перевірка прогнозу для інтервалів параметрів

Побудова матриці невідповідності (рис 5.2.4)

```
y_pred = best_classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, cmap="Purples", fmt="d")
plt.title("Confusion Matrix")
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.grid(False)
plt.show()
```

Рис. 5.2.4 Побудова матриці невідповідності

Тепер переглянемо результати виводу та графіки для групованих та усіх знижок окремо

```
Найкращі параметри:  KNeighborsClassifier(n_neighbors=2)
Точність KNN на тренувальній вибірці: 0.99636
Точність KNN на тестовій вибірці: 0.97826
----------------------------------------
```

Рис 5.2.5 Для групованих знижок

```
Best parameters:  KNeighborsClassifier(n_neighbors=2)
Точність KNN на тренувальній вибірці: 0.8561
Точність KNN на тестовій вибірці: 0.6087
```

Рис. 5.2.6 Для усіх знижок

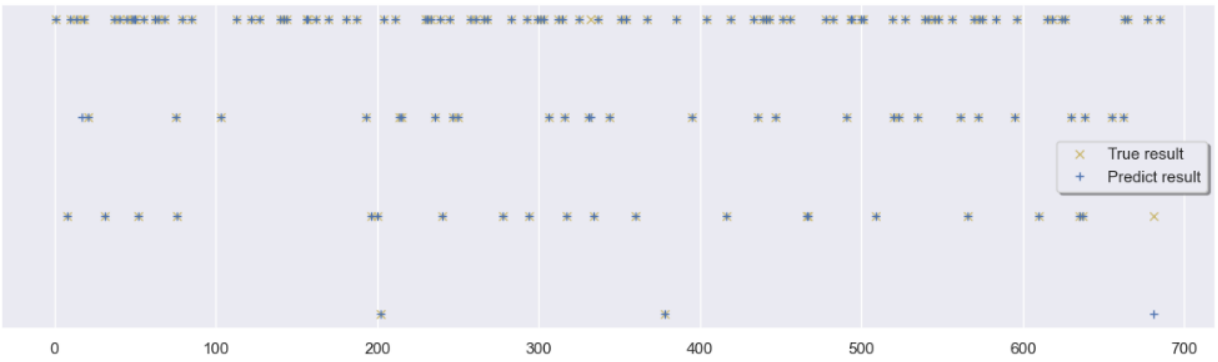


Рис. 5.2.7 Відображення прогнозу для інтервалів параметрів груповані знижки

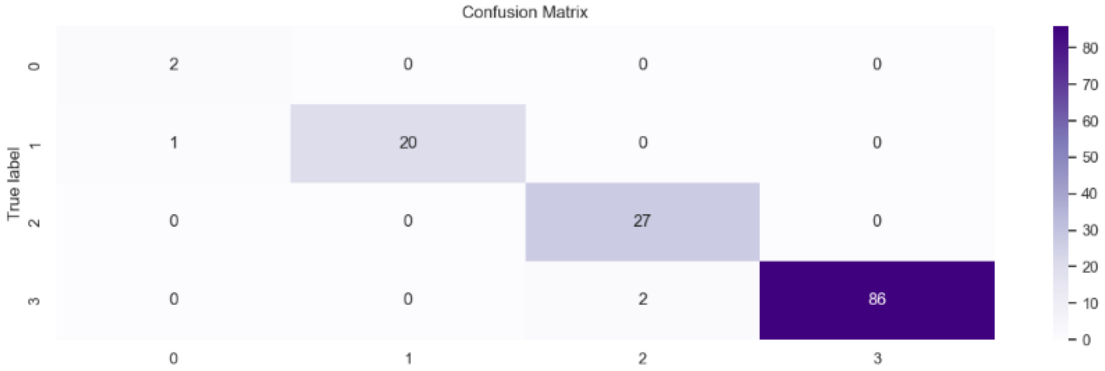


Рис. 5.2.8 Відображення матриці невідповідності груповані знижки

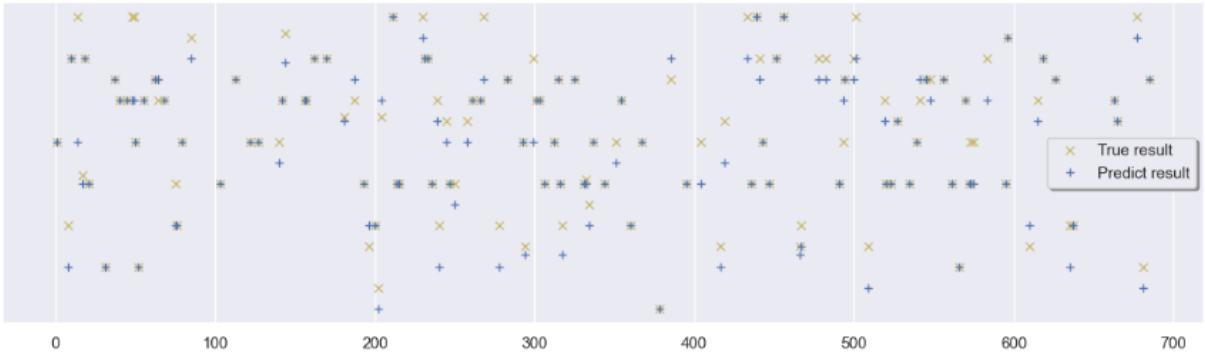


Рис. 5.2.9 Відображення прогнозу для інтервалів параметрів усіх знижок

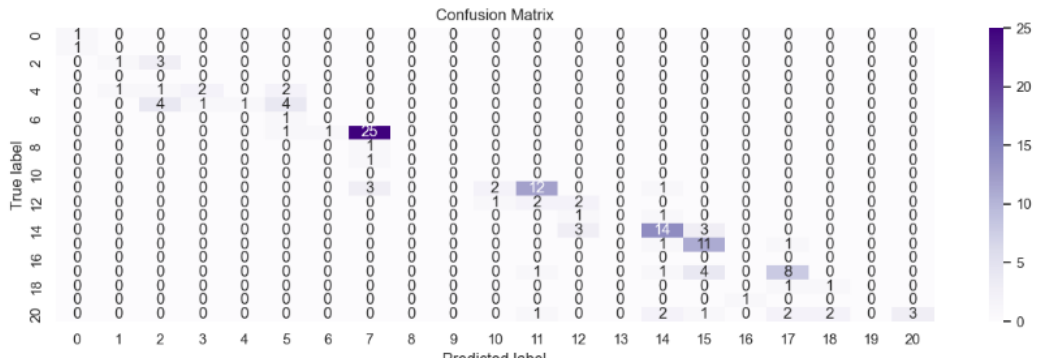


Рис. 5.2.10 Відображення матриці невідповідності усі знижки

Можемо зробиити висновки що при збільшенні класів зменшується точність.

Після аналізу результатів моделей KNN ми отримали наступні висновки:

1) Модель для групованих знижок:

- Найкращі параметри: KNeighborsClassifier(n_neighbors=2).
- Точність на тренувальній вибірці складає 99.636%.

- Точність на тестовій вибірці складає 97.826%.

2) Модель для усіх знижок:

- Найкращі параметри: KNeighborsClassifier(n_neighbors=2).
- Точність на тренувальній вибірці складає 85.61%.
- Точність на тестовій вибірці складає 60.87%.

Загалом, модель для чотирьох класів демонструє кращу точність як на тренувальній, так і на тестовій вибірці порівняно з моделлю для усіх знижок. Модель KNN з параметром n_neighbors=2 показала найкращі результати у прогнозуванні знижки на ігрові продукти. Матриця невідповідностей та графіки підтверджують високу продуктивність моделі для групованих знижок. В той же час, модель для усіх знижок має меншу точність і потребує подальшого вдосконалення для досягнення кращих результатів.

5.3 Аналіз отриманих результатів для методу Decision Tree Classifier

Для аналізу методу DTC підгонимо модель під тренувальні дані та перевіримо точність прогнозування отриманої модельки (рис 5.3.1).

```
dtc = DecisionTreeClassifier(random_state=0)
dtc.fit(X_train, y_train)

dtc_train_accuracy = round(dtc.score(X_train, y_train), 5)
dtc_test_accuracy = round(dtc.score(X_test, y_test), 5)

print("Точність DTC на тренувальній вибірці:", dtc_train_accuracy)
print("Точність DTC на тренувальній вибірці:", dtc_test_accuracy)
```

Рис 5.3.1 метод DTC

Також перевіримо прогноз для яких інтервалів параметрів виявився неправильним побудувавши графік(рис. 5.3.2)

```
plt.rcParams["figure.figsize"] = (15,4)
plt.gca().axes.get_yaxis().set_visible(False)
plt.plot(X_test.index, y_test, "yx", label = "True result")
plt.plot(X_test.index,dtc.predict(X_test), "b+", label = "Predict result")
plt.legend(loc='center right', shadow=True)
plt.show()
```

Рис 5.3.2 Прогнози методом DTC на інтервалах

```
y_pred = dtc.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, cmap="Blues", fmt="d")
plt.title("Confusion Matrix")
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.grid(False)
plt.show()
```

Рис. 5.3.3 Побудова матриці для DTC

Тепер переглянемо результати виводу та графіки для групованих та усіх знижок окремою

```
Точність DTC на тренувальній вибірці: 1.0
Точність DTC на тренувальній вибірці: 1.0
```

Рис 5.3.4 Для групованих знижок

```
Точність DTC на тренувальній вибірці: 1.0
Точність DTC на тренувальній вибірці: 0.75362
```

Рис 5.3.5 Для усіх знижок

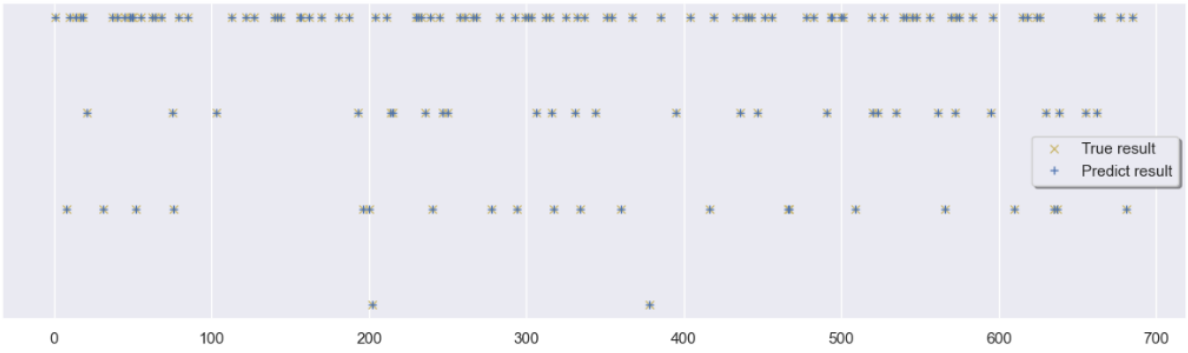


Рис. 5.3.6 Відображення прогнозу для інтервалів групованих знижок ’

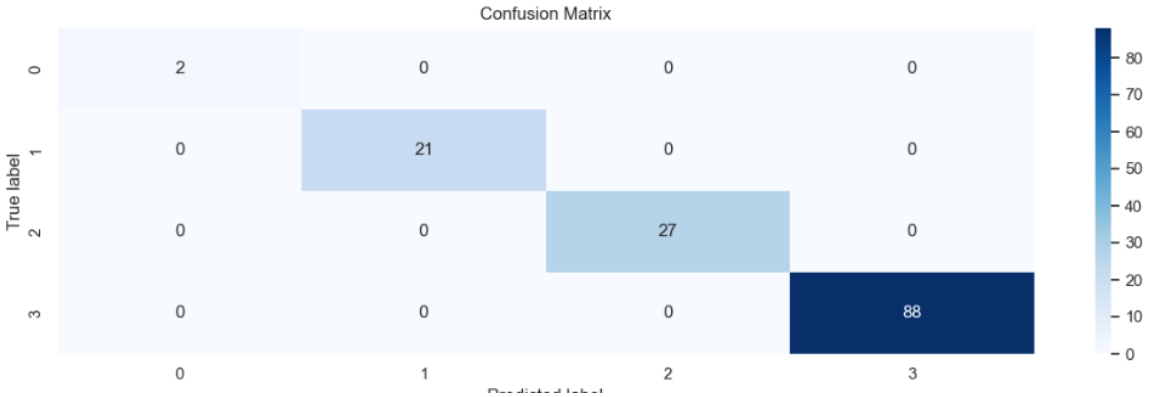


Рис. 5.3.7 Відображення матриці невідповідності груповані знижки

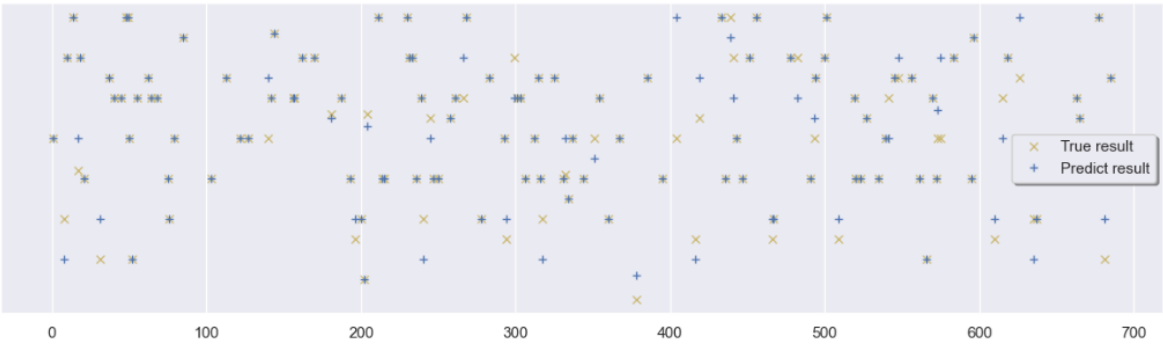


Рис. 5.3.8 Відображення прогнозу для інтервалів усіх знижок

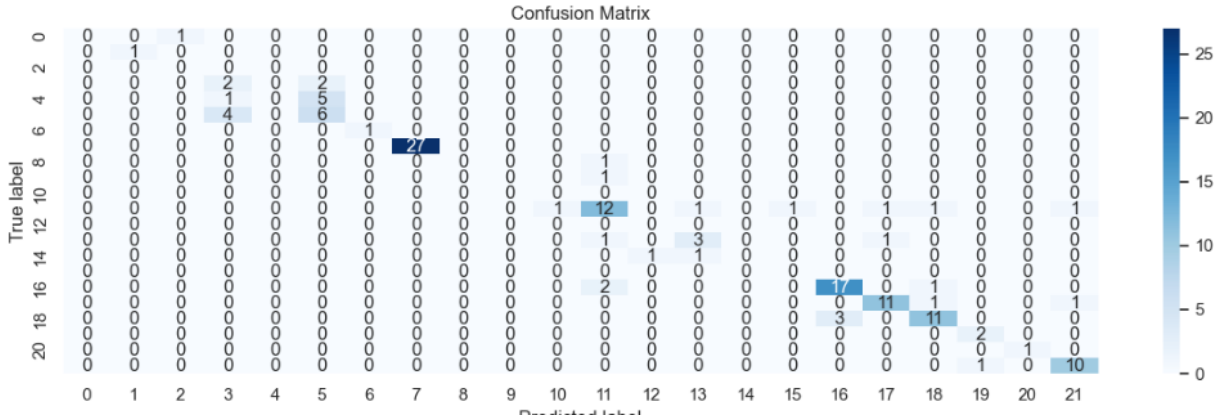


Рис. 5.3.9 Відображення матриці невідповідності груповані знижки

Після аналізу результатів моделі DTC (Decision Tree Classifier) ми отримали наступні висновки:

1) Модель DTC з групованими зниижками:

- Точність на тренувальній вибірці складає 100%.
- Точність на тестовій вибірці складає 100%.

Графік показує, що прогнозовані значення добре співпадають зі справжніми значеннями. Матриця невідповідностей підтверджує високу точність моделі, де всі значення класифікуються правильно.

2) Модель DTC з усіма знижками:

- Точність на тренувальній вибірці складає 100%.
- Точність на тестовій вибірці складає 75.362%.

Графік показує, що прогнозовані значення мають значні відхилення від справжніх значень. Матриця невідповідностей підтверджує меншу точність моделі, де деякі значення неправильно класифікуються.

Загалом, перший набір параметрів для групованих знижок моделі DTC демонструє високу точність і добре прогнозує значення, як на тренувальній, так і на тестовій вибірці.

5.4 Порівняння отриманих результатів

Для прийняття остаточного рішення щодо вибору найкращої моделі для подальшого прогнозування порівняємо результати(рис. 5.4.1), (додаток Б)

```
Аналіз 1
Точність на тренувальній вибірці:
KNN classifier:  99 %, DTC classifier:  100 %
Точність на тестовій вибірці:
KNN classifier:  97 %, DTC classifier:  100 %
----------------------------------------
Аналіз 2
Точність на тренувальній вибірці:
KNN classifier:  85 %, DTC classifier:  100 %
Точність на тестовій вибірці:
KNN classifier:  60 %, DTC classifier:  75 %
```

Рис. 5.4.1 Порівняння результатів «Аналіз 1» – для групованих знижок, Аналіз 1» – для усіх.

Також порівняємо матриці представлені на рисунках 5.2.8, 5.2.10 та 5.3.7, 5.3.9

Отже за результатами аналізу 1 ми бачимо, що обидві моделі, KNN (K-найближчих сусідів) і DTC (дерево класифікації рішень), показують високу точність на тренувальній вибірці. Точність KNN становить 99%, а точність DTC - 100%. На тестовій вибірці точність KNN становить 97%, а точність DTC - також 100%. У аналізі 2 ми бачимо, що точність KNN на тренувальній вибірці складає 85%, а точність DTC - 100%. Однак, на тестовій вибірці точність KNN знижується до 60%, в той час як точність DTC становить 75%. Робимо такі висновки:

У аналізі 1 обидві моделі показують дуже високу точність, проте DTC має перевагу, оскільки вона досягає 100% точності на тестовій вибірці.

У аналізі 2 KNN показує меншу точність на тестовій вибірці, тому DTC може бути кращим варіантом для прогнозування.

ВИСНОВКИ

Аналіз класифікації та прогнозування знижок на ігри для платформ PlayStation4 та 5 є важливим інструментом для геймерів та розробників. Знижки та спеціальні акції роблять ігри більш доступними, але при плануванні покупки виникає багато питань. Дана робота може допомогти відповісти на ці питання та прийняти обґрунтоване рішення, щодо покупки та економії коштів. У ній ми провели аналіз класифікації та прогнозування знижок на ігри для PlayStation4 та 5 з використанням методу найближчих сусідів (KNN) та класифікатора дерева рішень (DTC). Для цього використано набір даних, що містить інформацію про оригінальну ціну, знижку, рік випуску та інші параметри ігор.

Основна мета роботи полягала у розробці моделей, які можуть класифікувати знижки на ігри залежно від їх параметрів та прогнозувати величину знижки на основі історичних даних. Це дозволяє зрозуміти, які фактори впливають на розмір знижки та які ігри мають найбільші шанси на отримання значної знижки. Отримані моделі можуть бути корисними для розробників та видавництв ігор при стратегічному плануванні цін та знижок на свої продукти.

Аналіз класифікації та прогнозування знижок на ігри для PlayStation4 та 5 має великий потенціал у полі планування покупок ігор та розвитку галузі відеоігор в цілому. Це дозволить геймерам та шукачам вигідних пропозицій зекономити кошти та придбати більше ігор за доступною ціною.

ПЕРЕЛІК ПОСИЛАНЬ

1) https://pandas.pydata.org/docs/ - документація бібліотеки Pandas, яка надає потужні інструменти для обробки та аналізу даних.
2) https://scikit-learn.org/stable/index.html - документація бібліотеки scikit-learn, яка містить багато алгоритмів машинного навчання, метрики та інструменти для моделювання даних.
3) https://seaborn.pydata.org/documentation.html - документація бібліотеки seaborn, яка надає інструменти для статистичної візуалізації даних.
4) https://matplotlib.org/stable/users/index.html - документація бібліотеки matplotlib, яка дозволяє створювати різноманітні графіки і візуалізації даних.
5) https://numpy.org/doc/ - документація бібліотеки NumPy, яка надає підтримку для роботи з багатовимірними масивами та математичними функціями.
6) https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html - документація класу KNeighborsClassifier з бібліотеки scikit-learn, який реалізує алгоритм k-найближчих сусідів для класифікації.
7) https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html - документація класу GridSearchCV з бібліотеки scikit-learn, який дозволяє автоматично настроювати параметри моделей за допомогою хрестової перевірки.
8) https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html - документація класу DecisionTreeClassifier з бібліотеки scikit-learn, який реалізує алгоритм дерева рішень для класифікації.

ДОДАТОК А ТЕКСТИ ПРОГРАМНОГО КОДУ

Тексти програмного коду прогнозування доходів та факторів на популярність індустрії відеоігор

(Найменування програми (документа))

SSD

(Вид носія даних)

(Обсяг програми (документа), арк., Кб)

студента групи ІП-13 ІІ курсу
Романюк Діани Олексіївни

```
DROP EXTENSION IF EXISTS dblink;
CREATE EXTENSION dblink;

SELECT db.*
   INTO play_stat
   FROM dblink('dbname=k_stage user=admin password=228380', 'SELECT name,
platforms, original_price, discounted_price,
   discount, product_type, rank_in_sale, sale_name, date_recorded
   FROM k_stage.public.list1')
      AS db(name text,
           platforms text,
           original_price text,
           discounted_price text,
           discount text,
           product_type text,
           rank_in_sale integer,
           sale_name text,
           date_recorded text);
ALTER TABLE play_stat ADD COLUMN id integer GENERATED BY DEFAULT
AS IDENTITY PRIMARY KEY;

SELECT db.*
   INTO play_stat1_2
   FROM dblink('dbname=k_stage user=admin password=228380', 'SELECT name,
platforms, original_price, discounted_price,
   discount, product_type, rank_in_sale, sale_name, date_recorded
   FROM k_stage.public.list2')
      AS db(name text,
           platforms text,
           original_price text,
           discounted_price text,
           discount text,
           product_type text,
           rank_in_sale integer,
           sale_name text,
           date_recorded text);
ALTER TABLE play_stat1_2 ADD COLUMN id integer GENERATED BY
DEFAULT AS IDENTITY PRIMARY KEY;

SELECT db.*
   INTO play_stat2
   FROM dblink('dbname=k_stage user=admin password=228380', 'SELECT
"GameName", "Publisher", "ReleaseDate", "Developer", "Genre", "OfficialWebsite"
   FROM k_stage.public.alist')
      AS db(name text,
           publisher text,
```

```
        rel_date date,
        developer text,
        genre text,
        website text);
ALTER TABLE play_stat2 ADD COLUMN id integer GENERATED BY
DEFAULT AS IDENTITY PRIMARY KEY;

INSERT INTO play_stat(name, platforms, original_price, discounted_price, discount,
product_type, rank_in_sale,
                sale_name, date_recorded)
select distinct name, platforms, original_price, discounted_price, discount,
product_type, rank_in_sale,
            sale_name, date_recorded
FROM play_stat1_2;


INSERT INTO games(name, platform,product_type, publisher, developer, genre,
website)
select distinct play_stat.name, platforms, product_type, publisher, developer, genre,
website
FROM play_stat, play_stat2 WHERE play_stat.name = play_stat2.name order by
name;

DELETE FROM
games b
USING games a
where a.game_id < b.game_id and a.name = b.name;

INSERT INTO discounts(sale_name, discounted_price, discount, game,
rank_in_sale)
select distinct sale_name, discounted_price, discount, name, rank_in_sale
FROM play_stat;


INSERT INTO tdates (day, month, year)
SELECT
  EXTRACT(DAY FROM parsed_date) AS day,
  EXTRACT(MONTH FROM parsed_date) AS month,
  EXTRACT(YEAR FROM parsed_date) AS year
FROM (
  SELECT TO_DATE(rel_date::text, 'YYYY-MM-DD') AS parsed_date
  FROM play_stat2
) AS subquery;
DELETE FROM tdates
WHERE day IS NULL AND month IS NULL AND year IS NULL;
```

```
DELETE FROM
tdates b
USING tdates a
where a.date_id < b.date_id and a.day = b.day and a.month = b.month and a.year =
b.year;

insert into dates(day, month, year) SELECT day, month, year from tdates;

DELETE FROM
play_stat b
USING play_stat a
where a.id < b.id and a.name = b.name;

insert into f_prices(game_id, price)
SELECT
   g.game_id AS game_id,
   ps.original_price AS price
FROM  play_stat ps
JOIN games g ON ps.name = g.name;

INSERT INTO new_dataset(name, platforms, original_price, discounted_price,
discount, product_type, publisher, rel_date,
                developer, genre)
select distinct play_stat.name, platforms, original_price, discounted_price, discount,
product_type, publisher, rel_date,
                developer, genre
FROM play_stat, play_stat2 where play_stat.name = play_stat2.name;

INSERT INTO sales_f_stat(game_id, disc_id, date_id)
SELECT
  g.game_id AS game_id,
  d.disc_id AS discount_id,
  dt.date_id as date_id
FROM new_dataset n
LEFT JOIN discounts d ON n.name = d.game
LEFT JOIN games g ON n.name = g.name
LEFT JOIN dates dt ON TO_DATE(n.rel_date::text, 'YYYY-MM-DD') =
TO_DATE(CONCAT(dt.day, '-', dt.month, '-', dt.year), 'dd-mm-yyyy')
order by g.name;

INSERT INTO sales_f_stat(game_id, disc_id, date_id)
SELECT
  g.game_id AS game_id,
  d.disc_id AS discount_id,
  dt.date_id as date_id
FROM new_dataset n
```

```
JOIN discounts d ON n.name = d.game
JOIN games g ON n.name = g.name
JOIN dates dt ON TO_DATE(n.rel_date::text, 'YYYY-MM-DD') =
TO_DATE(CONCAT(dt.day, '-', dt.month, '-', dt.year), 'dd-mm-yyyy')
order by g.name;

DELETE FROM new_dataset
WHERE name IS NULL
   OR platforms IS NULL
   OR original_price IS NULL
   OR discounted_price IS NULL
   OR discount IS NULL
   OR product_type IS NULL
   OR publisher IS NULL
   OR rel_date IS NULL
   OR developer IS NULL
   OR genre IS NULL;

ALTER TABLE new_dataset
ALTER COLUMN original_price TYPE numeric
USING replace(substring(original_price, 2), ',', '.')::numeric;

UPDATE new_dataset
SET discounted_price = NULL
WHERE discounted_price = 'Included';

ALTER TABLE new_dataset
ALTER COLUMN discounted_price TYPE numeric
USING replace(substring(discounted_price, 2), ',', '.')::numeric;

UPDATE new_dataset
SET discount = CAST(REPLACE(discount, '%', '') AS numeric)
WHERE discount LIKE '%';

ALTER TABLE new_dataset RENAME COLUMN discount TO "discount(%)";
ALTER TABLE new_dataset RENAME COLUMN original_price TO
"original_price(£)";
ALTER TABLE new_dataset RENAME COLUMN discounted_price TO
"discounted_price(£)";
```

ДОДАТОК Б ТЕКСТИ ПРОГРАМНОГО КОДУ

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix

import seaborn as sns
import matplotlib.pyplot as plt


pd.set_option('display.max_columns', None)
dataframe = pd.read_csv('data.csv', delimiter=',', decimal=',', encoding='UTF-8')
dataframe.info()
print(dataframe.head(5))

dataframe.dropna(inplace=True)  # Видалення рядків з нульовими значеннями

dataframe['discounted_price(£)'] = dataframe['discounted_price(£)'].astype(float)
dataframe['original_price(£)'] = dataframe['original_price(£)'].astype(float)
dataframe['discount(%)'] = dataframe['discount(%)'].astype(float)
dataframe['discount(%)'] = dataframe['discount(%)'].abs()

dataframe.info()

sns.set(style='darkgrid')
plt.figure(figsize=(12, 6))

colors = sns.color_palette("Purples", len(dataframe['discount(%)'].unique()))

sns.countplot(x='discount(%)', data=dataframe, palette=colors)
plt.xlabel('Discount (%)')
plt.ylabel('Count')
plt.title('Distribution of Discounts')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()

dataframe['rel_date'] = pd.to_datetime(dataframe['rel_date'], format='%Y-%m-%d')
dataframe['year'] = dataframe['rel_date'].dt.year.astype(float)
print(dataframe.head(5))

dataframe['discount_label'] = dataframe['discount(%)'].apply(lambda x: 0 if x < 30
else (1 if x < 50 else (2 if x == 50 else 3)))
dataframe.info()
```

```
data2 = dataframe.loc[:, ['discount(%)', 'year', 'original_price(£)',
'discounted_price(£)', 'discount_label']]
#-----------------------------------------------
print("-----------------------------------------")

sns.set(style='darkgrid')
plt.figure(figsize=(12, 6))

colors = sns.color_palette("Blues", len(data2['discount_label'].unique()))

sns.countplot(x='discount_label', data=data2, palette=colors)
plt.xlabel('Discount_label')
plt.ylabel('Count')
plt.title('Distribution of Discounts')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()

#-----------------------------------------
#Перший аналіз NKK

# Розділення даних на ознаки (X) та цільову змінну (y)
X = data2.drop(columns='discount_label')
y = data2['discount_label']

# Розділення на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8,
random_state=0)

# Ініціалізація KNN-класифікатора та параметрів для пошуку
classifier = KNeighborsClassifier()
parameters = {'n_neighbors': range(1, 25)}
grid_search = GridSearchCV(classifier, parameters, cv=10, verbose=1)
grid_search.fit(X_train, y_train)
print("Найкращі параметри: ", grid_search.best_estimator_)

# Використання найкращих параметрів для навчання моделі
best_classifier = grid_search.best_estimator_
best_classifier.fit(X_train, y_train)

# Прогнозування на тренувальній та тестовій вибірці
train_accuracy = round(best_classifier.score(X_train, y_train), 5)
test_accuracy = round(best_classifier.score(X_test, y_test), 5)

print("Точність KNN на тренувальній вибірці:", train_accuracy)
```

```
print("Точність KNN на тестовій вибірці:", test_accuracy)

plt.rcParams["figure.figsize"] = (15,4)
plt.gca().axes.get_yaxis().set_visible(False)
plt.plot(X_test.index, y_test, "yx", label = "True result")
plt.plot(X_test.index, best_classifier.predict(X_test), "b+", label = "Predict result")
plt.legend(loc='center right', shadow=True)
plt.show()

y_pred = best_classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, cmap="Purples", fmt="d")
plt.title("Confusion Matrix")
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.grid(False)
plt.show()

compNKK1train = int(best_classifier.score(X_train, y_train)*100)
compNKK1test = int(best_classifier.score(X_test, y_test)*100)
KNN1 = best_classifier

print("-----------------------------------------")
#-----------------------------------------------
#аналіз 2 NKK

X = data2.drop(columns='discount(%)')
y = data2['discount(%)']

# Розділення на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8,
random_state=0)

# Ініціалізація KNN-класифікатора та параметрів для пошуку
classifier = KNeighborsClassifier()
parameters = {'n_neighbors': range(2, 25)}
grid_search = GridSearchCV(classifier, parameters, cv=10, verbose=1)
grid_search.fit(X_train, y_train)
print("Best parameters: ", grid_search.best_estimator_)

# Використання найкращих параметрів для навчання моделі
best_classifier = grid_search.best_estimator_
best_classifier.fit(X_train, y_train)

# Прогнозування на тренувальній та тестовій вибірці
```

```
train_accuracy = round(best_classifier.score(X_train, y_train), 5)
test_accuracy = round(best_classifier.score(X_test, y_test), 5)

print("Точність KNN на тренувальній вибірці:", train_accuracy)
print("Точність KNN на тестовій вибірці:", test_accuracy)

plt.rcParams["figure.figsize"] = (15,4)
plt.gca().axes.get_yaxis().set_visible(False)
plt.plot(X_test.index, y_test, "yx", label = "True result")
plt.plot(X_test.index, best_classifier.predict(X_test), "b+", label = "Predict result")
plt.legend(loc='center right', shadow=True)
plt.show()

y_pred = best_classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, cmap="Purples", fmt="d")
plt.title("Confusion Matrix")
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.grid(False)
plt.show()

compNKK2train = int(best_classifier.score(X_train, y_train)*100)
compNKK2test = int(best_classifier.score(X_test, y_test)*100)
KNN2 = best_classifier

print("-----------------------------------------")
#-------------------------------------------------
#аналіз 1 DTC
X = data2.drop(columns='discount_label')
y = data2['discount_label']

# Розділення на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8,
random_state=0)

dtc = DecisionTreeClassifier(random_state=0)
dtc.fit(X_train, y_train)

dtc_train_accuracy = round(dtc.score(X_train, y_train), 5)
dtc_test_accuracy = round(dtc.score(X_test, y_test), 5)

print("Точність DTC на тренувальній вибірці:", dtc_train_accuracy)
print("Точність DTC на тренувальній вибірці:", dtc_test_accuracy)
```

```
plt.rcParams["figure.figsize"] = (15,4)
plt.gca().axes.get_yaxis().set_visible(False)
plt.plot(X_test.index, y_test, "yx", label = "True result")
plt.plot(X_test.index,dtc.predict(X_test), "b+", label = "Predict result")
plt.legend(loc='center right', shadow=True)
plt.show()

y_pred = dtc.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, cmap="Blues", fmt="d")
plt.title("Confusion Matrix")
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.grid(False)
plt.show()

compCTD1train = int(dtc.score(X_train, y_train)*100)
compCTD1test = int(dtc.score(X_test, y_test)*100)
DTC1 = dtc

print("-----------------------------------------")
#-------------------------------------------------
#аналіз 2 DTC
X = data2.drop(columns='discount(%)')
y = data2['discount(%)']

# Розділення на тренувальну та тестову вибірки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, train_size=0.8,
random_state=0)

dtc = DecisionTreeClassifier(random_state=0)
dtc.fit(X_train, y_train)

dtc_train_accuracy = round(dtc.score(X_train, y_train), 5)
dtc_test_accuracy = round(dtc.score(X_test, y_test), 5)

print("Точність DTC на тренувальній вибірці:", dtc_train_accuracy)
print("Точність DTC на тренувальній вибірці:", dtc_test_accuracy)

plt.rcParams["figure.figsize"] = (15,4)
plt.gca().axes.get_yaxis().set_visible(False)
plt.plot(X_test.index, y_test, "yx", label = "True result")
plt.plot(X_test.index,dtc.predict(X_test), "b+", label = "Predict result")
plt.legend(loc='center right', shadow=True)
plt.show()
```

```
y_pred = dtc.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

sns.heatmap(cm, annot=True, cmap="Blues", fmt="d")
plt.title("Confusion Matrix")
plt.xlabel("Predicted label")
plt.ylabel("True label")
plt.grid(False)
plt.show()
compCTD2train = int(dtc.score(X_train, y_train)*100)
compCTD2test = int(dtc.score(X_test, y_test)*100)
DTC2 = dtc

print("-----------------------------------------")
print("Аналіз 1")
print("Точність на тренувальній вибірці:")
print("KNN classifier: ", compNKK1train, '%,', 'DTC classifier: ', compCTD1train,
'%')
print("Точність на тестовій вибірці:")
print("KNN classifier: ", compNKK1test, '%,', 'DTC classifier: ', compCTD1test, '%')

print("-----------------------------------------")
print("Аналіз 2")
print("Точність на тренувальній вибірці:")
print("KNN classifier: ", compNKK2train, '%,', 'DTC classifier: ', compCTD2train,
'%')
print("Точність на тестовій вибірці:")
print("KNN classifier: ", compNKK2test, '%,', 'DTC classifier: ', compCTD2test, '%')
```