

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно- обчислювальних систем і управління

ОНИСЬКІВ Олександр Васильович

**Інтелектуальний застосунок аналізу впливу візуального виконання
комп'ютерних ігор на ринковий успіх / An intelligent application for analyzing
the impact of computer game visual performance on market success**

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Комп'ютерні науки
Кваліфікаційна робота

Виконав студент групи КН-43
О.В. Ониськів

Науковий керівник:
к.т.н., доцент, М.З. Домбровський

Кваліфікаційну роботу

допущено до захисту:

" ____ " _____ 20 ____ р.

Завідувач кафедри

_____ Н. В. Дзюбановська

ТЕРНОПІЛЬ – 2026

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність: 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.В. Дзюбановська
« ____ » _____ 20__ р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
ОНИСЬКІВУ Олександр Васи́льовичу**

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи

Інтелектуальний застосунок аналізу впливу візуального виконання компютерних ігор на ринковий успіх / An intelligent application for analyzing the impact of computer game visual performance on market success

керівник роботи к.т.н., доцент, М.З. Домбровський

затверджений наказом по університету від 01 грудня 2025 року № 894.

2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2026 р.

3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити

- дослідити існуючі підходи до оцінювання візуального виконання компютерних систем ігрового типу, визначити фактори що впливають на їх ринковий успіх;
- розробити алгоритм вилучення візуальних характеристик з ігрових матеріалів;
- розробити та валідувати застосунок, що викоритсовує прогностичну модель ринкового успіху на основі аналізу візуальних характеристик;
- реалізувати програмний застосунок з графічним інтерфейсом користувача;
- провести тестування розробленого програмного продукту

5. Перелік графічного матеріалу у роботі

- Структурна схема програмного застосунку.
- Алгоритм обробки візуальних даних.
- ER-діаграма база даних.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 01 грудня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Огляд літературних джерел відповідно до предметної області та складання плану роботи	до 01.01.2026 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02.2026 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 15.03.2026 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05.2026 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2026 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2026 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2026 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2026 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06.2026 р.	

Студент _____ О.В. Ониськів
підпис

Керівник роботи _____ к.т.н., доцент М.З. Домбровський
підпис

АНОТАЦІЯ

Кваліфікаційна робота на тему «Інтелектуальний застосунок аналізу впливу візуального виконання комп'ютерних ігор на ринковий успіх» на здобуття освітнього ступеня «бакалавр» зі спеціальності 122 «Комп'ютерні науки» освітньо-професійної програми «Комп'ютерні науки» виконана обсягом 64 сторінок і містить 13 ілюстрацій, 7 таблиць, 3 додатки та 32 використаних джерел.

Метою роботи є розроблення інтелектуального застосунку для автоматизованого аналізу залежності між якісними характеристиками візуального виконання комп'ютерних ігор та їхнім ринковим успіхом на основі методів машинного навчання і статистичного аналізу.

Для досягнення мети використано методи кореляційно-регресійного аналізу, методи комп'ютерного зору та обробки зображень, методи класифікації та кластеризації даних, а також методи порівняльного аналізу програмних рішень.

У результаті виконання роботи розроблено програмний застосунок, що дозволяє автоматично збирати та обробляти дані про візуальні характеристики ігор, будувати прогностичні моделі комерційного успіху на основі цих характеристик та візуалізувати результати аналізу для прийняття рішень.

Результати дослідження можуть бути використані розробниками ігор, видавцями, аналітичними агентствами та інвестиційними компаніями для оцінки ринкового потенціалу продуктів на ранніх стадіях розробки.

Ключові слова: КОМП'ЮТЕРНІ ІГРИ, ВІЗУАЛЬНИЙ АНАЛІЗ, МАШИННЕ НАВЧАННЯ, РИНКОВИЙ УСПІХ, ОБРОБКА ЗОБРАЖЕНЬ, КОРЕЛЯЦІЙНИЙ АНАЛІЗ, STEAM, ПРОГНОСТИЧНА МОДЕЛЬ.

ANNOTATION

Qualification work on the topic "Intellectual Application for Analysis of the Impact of Computer Games Visual Performance on Market Success" for the Bachelor's degree in specialty 122 "Computer Science" is written on 64 pages and contains 13 figures, 7 tables, 3 appendices and 32 sources.

The purpose of the work is to develop an intelligent application for automated analysis of the relationship between qualitative characteristics of the visual performance of computer games and their market success based on machine learning methods and statistical analysis.

Methods of correlation-regression analysis, computer vision and image processing methods, data classification and clustering methods, and comparative analysis of software solutions were used to achieve the goal.

As a result of the work, a software application was developed that allows automatic collection and processing of data on visual game characteristics, construction of predictive models of commercial success based on these characteristics, and visualization of analysis results for decision-making.

The research results can be used by game developers, publishers, analytics agencies and investment companies to assess the market potential of products at early stages of development.

Keywords: COMPUTER GAMES, VISUAL ANALYSIS, MACHINE LEARNING, MARKET SUCCESS, IMAGE PROCESSING, CORRELATION ANALYSIS, STEAM, PREDICTIVE MODEL.

ЗМІСТ

Перелік умовних позначень	7
Вступ.....	8
1 Аналіз предметної області та огляд існуючих рішень	10
1.1 Ринок комп'ютерних ігор та фактори комерційного успіху	10
1.2 Роль візуального виконання у просуванні ігор	14
1.3 Огляд існуючих аналітичних інструментів та їх недоліки	17
1.4 Постановка завдання дослідження	19
2 Алгоритмічне та інформаційне забезпечення системи	21
2.1 Загальна архітектура застосунку	21
2.2 Методи обробки візуальних даних.....	22
2.3 Алгоритм побудови прогностичної моделі	24
2.4 Інформаційне забезпечення системи.....	26
3 Програмно-технологічне забезпечення системи.....	29
3.1 Реалізація програмного забезпечення	29
3.2 Інтерфейс користувача	37
3.3 Тестування розробленої програми	44
Висновки	48
Список використаних джерел	50
Додаток А Копія опублікованих результатів	53
Додаток Б Фрагмент коду модуля VisualAnalyzer.....	56
Додаток В SQL-скрипт створення таблиць бази даних	64

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface – інтерфейс програмування застосунків

CNN – Convolutional Neural Network – згорткова нейронна мережа

GUI – Graphical User Interface – графічний інтерфейс користувача

ML – Machine Learning – машинне навчання

МН – машинне навчання

ПЗ – програмне забезпечення

ПС – програмна система

БД – база даних

КН – комп'ютерні науки

ІС – інформаційна система

ВСТУП

Індустрія комп'ютерних ігор є одним із найдинамічніших секторів цифрової економіки: за даними аналітичної компанії Newzoo, обсяг світового ринку ігор перевищив 200 млрд доларів США у 2023 році і продовжує зростати [1]. Щорічно на платформі Steam виходить понад 10 000 нових ігрових проєктів, що породжує жорстку конкуренцію і підвищує значущість прийняття обґрунтованих маркетингових і технічних рішень на всіх етапах розробки.

Серед численних факторів, що визначають комерційний успіх комп'ютерної гри, особливе місце посідає візуальне виконання. Під цим терміном розуміють сукупність характеристик графічного рендерингу, художнього стилю, якості текстур, анімацій і загальної естетики продукту. Попередні дослідження свідчать про існування значущого кореляційного зв'язку між метриками візуальної якості та рейтингами, рецензіями і обсягами продажів ігрових продуктів [2, 3]. Однак більшість наявних підходів до оцінювання цього зв'язку залишаються переважно якісними й суб'єктивними.

Актуальність теми зумовлена необхідністю автоматизації процесу аналізу візуальних характеристик ігор та кількісного оцінювання їхнього впливу на ринковий успіх. Розробка інтелектуального застосунку, здатного з великих обсягів зображень і відеоматеріалів, вилучати релевантні метрики та будувати прогностичні моделі, дозволить значно скоротити час і витрати на маркетинговий аналіз.

Об'єкт роботи є процеси збору, представлення, обробки, зберігання, передачі та доступу до інформації в комп'ютерних системах.

Предмет роботи є інформаційні процеси інтелектуального аналізу впливу характеристик візуального виконання щодо ринкового успіху комп'ютерних ігор на цифрових платформах.

Метою кваліфікаційної роботи є прогнозування комерційного та ринкового успіху візуального виконання комп'ютерних ігор на основі розроблення інтелектуального застосунку.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- дослідити існуючі підходи до оцінювання візуального виконання комп'ютерних систем ігрового типу, визначити фактори що впливають на їх ринковий успіх;
- розробити алгоритм вилучення візуальних характеристик з ігрових матеріалів;
- розробити та валідувати застосунок, що використовує прогностичну модель ринкового успіху на основі аналізу візуальних характеристик;
- реалізувати програмний застосунок з графічним інтерфейсом користувача;
- провести тестування розробленого програмного продукту.

Методами дослідження є: методи кореляційно-регресійного аналізу, методи машинного навчання (зокрема, методи градієнтного бустингу та регресійні ансамблеві підходи), методи комп'ютерного зору та обробки зображень на основі бібліотеки OpenCV, методи порівняльного аналізу програмних рішень.

Практичне значення роботи полягає у тому, що розроблений застосунок може бути застосований розробниками ігор для раннього виявлення потенційних проблем візуального виконання, видавцями для оцінки ринкового потенціалу проектів, а також аналітичними агентствами для формування галузевих звітів.

Основні результати кваліфікації роботи були представлені та опубліковані на II Міжнародній науково-практичній конференції “Сталий розвиток економіки та суспільства”, м. Суми, 15 червня 2026 р (Додаток А).

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

1.1 Ринок комп'ютерних ігор та фактори комерційного успіху

Сучасна індустрія комп'ютерних ігор є складною та багатогранною екосистемою, що охоплює мобільні, консольні та PC-платформи, кожна з яких має свою унікальну економічну модель, аудиторію та технічні особливості [3]. Мобільний сегмент, який включає ігри для смартфонів та планшетів на операційних системах iOS та Android, залишається найбільшим за кількістю користувачів, охоплюючи мільярди гравців по всьому світу [3, 4]. Консольний ринок, представлений платформами Sony PlayStation, Microsoft Xbox та Nintendo Switch, характеризується високою лояльністю аудиторії та ексклюзивними проектами, що стають драйверами продажів апаратних засобів [4]. PC-геймінг, у свою чергу, пропонує найбільшу гнучкість у налаштуваннях, підтримку модифікацій та доступ до численних магазинів цифрової дистрибуції [5].

Сукупні доходи галузі формуються з кількох основних джерел: продажу копій ігор (як фізичних, так і цифрових), мікротранзакцій (внутрішньоігрових покупок косметичних предметів, бустерів, валюти), підписок (таких як Xbox Game Pass, PlayStation Plus, EA Play) і рекламних надходжень (особливо актуальних для мобільних free-to-play проектів). За прогнозами аналітичної компанії Statista, до 2026 року загальний ринок ігор досягне 260 млрд доларів США, демонструючи стійке зростання навіть попри економічні виклики, а частка цифрових продажів перевищить 95%, що свідчить про майже повний перехід галузі на цифрову дистрибуцію та згортання виробництва фізичних носіїв [4].

Платформа Steam компанії Valve залишається домінуючим майданчиком цифрової дистрибуції PC-ігор, фактично встановлюючи стандарти для всієї індустрії. Станом на початок 2024 року в каталозі Steam налічується понад 70000 ігор, а щомісячна активна аудиторія перевищує 130 мільйонів користувачів, що робить її наймасовішою платформою для PC-геймерів [5]. Успіх Steam зумовлений не лише широкою бібліотекою контенту, але й розвиненою інфраструктурою: системою досягнень, хмарними збереженнями, майстернею Steam Workshop для

модів, системою відгуків та рекомендацій, а також регулярними розпродажами, що стимулюють покупки. Відкритість платформи і доступність інструментів публікації через програму Steam Direct, яка вимагає від розробників лише сплати невеликого збору та дотримання базових вимог до контенту, призвели до різкого збільшення кількості нових виходів – щороку публікуються тисячі нових ігор. Це явище, відоме як "перенасичення ринку" (market saturation), створює серйозні виклики для розробників, особливо незалежних студій, оскільки ускладнює завдання виділення продукту серед конкурентів. У таких умовах якість візуальної презентації, маркетингова активність та вміння привернути увагу цільової аудиторії в перші ж дні після релізу стають критично важливими факторами виживання проекту.

Проведений аналіз джерел, дає змогу виокремити декілька груп факторів, що впливають на комерційний успіх комп'ютерної гри, кожен з яких вносить свій внесок у кінцевий результат. Перша група – це ігровий процес (gameplay), який охоплює механіки взаємодії гравця зі світом гри, рівень складності, баланс, систему прогресії та реіграбельність (можливість отримувати задоволення від повторного проходження). Ці характеристики традиційно вважаються визначальними для довгострокового утримання аудиторії, оскільки саме якісний геймплей змушує гравців повертатися до гри знову і знову, рекомендувати її друзям та формувати лояльне ком'юніті [6].

Друга група факторів – візуальна презентація, яка включає графічну якість (рівень деталізації, технічну досконалість), художній стиль (унікальність візуальної мови, відповідність жанру), якість анімацій та загальну естетику. Дослідження [7, 8] демонструють, що перше враження від скріншотів і трейлерів суттєво впливає на рішення про покупку, оскільки саме візуальний контент є першим, з чим стикається потенційний покупець у цифровому магазині. У умовах, коли користувач гортає стрічку нових релізів, у нього є лише кілька секунд, щоб зацікавитися грою, і якісна візуальна презентація стає вирішальним фактором кліку.

Третя група – маркетинг і PR-активність, що охоплює бюджет на рекламу (платні кампанії в соціальних мережах, на ігрових порталах, через інфлюенсерів), роботу з лідерами думок (стрімерами, ютуберами, блогерами), присутність у соціальних мережах та ком'юніті-менеджмент. Дослідження Clement [9] показало, що витрати на маркетинг корелюють з першотижневими продажами більш ніж на 70%, що підкреслює важливість інвестицій у просування, особливо на старті, коли алгоритми платформ формують початкову видимість продукту. Четверта група факторів – репутація розробника і бренд, яка включає історію попередніх релізів студії, наявність нагород, медійний образ та загальне сприйняття бренду аудиторією.

Серійні ігри (сіквели, франшизи) мають значно нижчий маркетинговий ризик, оскільки вже мають впізнаваність та лояльну базу фанатів, що гарантує певний рівень продажів ще до початку активної реклами [10]. П'ята група – технічні характеристики, такі як вимоги до заліза (мінімальні та рекомендовані системні вимоги), наявність підтримки актуальних API (DirectX, Vulkan) та технологій (трасування променів, FSR, DLSS тощо). Гравці з більш потужними системами готові платити преміум-ціну за ігри з передовою графікою, що використовує всі можливості сучасного заліза, тоді як погана оптимізація може призвести до хвилі негативних відгуків навіть при якісному контенті [11]. Шоста група – ціна та монетизаційна модель, що включає вартість придбання (стратегії ціноутворення, регіональне ціноутворення), наявність завантажувального контенту (DLC), модель free-to-play (безкоштовний вхід з монетизацією всередині), battle pass (сезонні перепустки) та інші механізми отримання доходу.

Таблиця 1.1 узагальнює значення кожного фактора відповідно до результатів метааналізу 47 академічних та галузевих досліджень, надаючи кількісну оцінку відносної важливості кожного елемента. На основі цих даних видно, що ігровий процес залишається найважливішим фактором з вагою 0,82, що підтверджує традиційне уявлення про пріоритет якості геймплею. Візуальна презентація посідає друге місце з вагою 0,74, що лише незначно поступається геймплею та перевищує маркетинг (0,71) та цінову політику (0,69). Це свідчить про те, що візуальна

складова є критично важливою для комерційного успіху, особливо на етапі залучення уваги потенційних покупців.

Таблиця 1.1 – Фактори комерційного успіху комп'ютерних ігор

Фактор	Вага (0–1)	Джерело
Ігровий процес	0,82	[6, 7]
Візуальна презентація	0,74	[8, 12]
Маркетинг і PR	0,71	[9, 13]
Репутація розробника	0,65	[10, 14]
Технічні характеристики	0,58	[11, 15]
Ціна та монетизація	0,69	[16, 17]

Наведені дані підтверджують, що візуальна презентація посідає друге місце за значущістю серед ключових факторів успіху, що робить її пріоритетним об'єктом для дослідження та оптимізації. Водночас, на відміну від ігрового процесу, який є суб'єктивним та складним для формалізації поняттям, візуальні характеристики піддаються кількісній оцінці методами комп'ютерного зору, що відкриває можливості для їх автоматизованого аналізу. Кольорові гістограми, текстурні метрики, ентропія зображення, розпізнавання об'єктів та стилів – усі ці інструменти дозволяють перетворити суб'єктивне враження від графіки на набір числових даних, придатних для статистичного аналізу та побудови прогностичних моделей.

Важливим аспектом є розуміння того, що ринковий успіх гри не є монолітною концепцією, яку можна виміряти одним універсальним показником. Залежно від контексту, цілей дослідження та бізнес-моделі, під успіхом можуть розумітися різні метрики: обсяг продажів (кількість проданих копій або загальний дохід), середній рейтинг критиків (Metacritic score, що агрегує оцінки професійних журналістів), рейтинг гравців (Steam user review score, що відображає задоволеність кінцевої аудиторії), пікова онлайн-аудиторія (кількість одночасних гравців, що свідчить про популярність та активність ком'юніті), кількість відгуків (як індикатор залученості гравців), час утримання гравця (playtime retention, що показує,

наскільки гра здатна захопити увагу на довгий термін). У цій роботі як основний показник успіху обрано комплексний індекс, що поєднує нормалізовані значення Steam-рейтингу та оцінений обсяг продажів за методологією SteamSpy [18]. Такий підхід дозволяє врахувати як якість гри з точки зору користувачів, так і її комерційну ефективність, надаючи більш збалансовану та об'єктивну оцінку успіху, ніж будь-яка окрема метрика.

У межах цього дослідження комп'ютерна гра розглядається не як статичний медіаресурс, а як складна інтегрована комп'ютерна система ігрового типу [1]. Вона функціонує як цілісна структура, що об'єднує чотири взаємозалежні підсистеми: графічну (відповідає за рендеринг, освітлення та візуальний стиль), логіко-алгоритмічну (ігрова логіка й правила), інтерактивну (елементи GUI та взаємодія з користувачем) і підсистему керування системними ресурсами [2].

Системний підхід дозволяє трактувати ринковий успіх грального проєкту як інтегральний результат ефективної синергії всіх її складових [3]. Оскільки графічна підсистема є першою точкою контакту користувача із комп'ютерною системою, аналіз її характеристик за допомогою обчислювальних алгоритмів є критично важливим для прогнозування комерційної життєздатності всього продукту [4].

1.2 Роль візуального виконання у просуванні ігор

Зі зростанням конкуренції на ринку цифрових ігор першорядного значення набуває здатність продукту привернути увагу потенційного покупця за мінімальний час. Маркетингові дослідження свідчать, що середня тривалість перегляду сторінки гри в магазині Steam перед рішенням про покупку або закриття не перевищує 15 секунд [19]. За цей час візуальні матеріали – скріншоти, трейлери та ключовий арт – відіграють вирішальну роль.

Поняття «візуальне виконання» охоплює декілька рівнів сприйняття. На технічному рівні оцінюються такі параметри, як роздільна здатність текстур, кількість полігонів, якість освітлення та тіней, ефекти частинок. На художньому

рівні значущими є стильова цілісність, кольорова палітра, композиція кадру. На рівні UX розглядається читаність інтерфейсу, контрастність та ергономіка.

Теоретичним підґрунтям для аналізу візуальних характеристик є методи комп'ютерного зору та цифрової обробки зображень, що дозволяють оцінювати кольорові, текстурні та структурні особливості графічного контенту. Модель GOMS (Goals, Operators, Methods, and Selection Rules) пояснює механізми прийняття рішень при взаємодії з графічним контентом [20]. Концепція «візуального шуму» (visual clutter) описує негативний вплив надмірної деталізації або хаотичного розташування елементів на сприйняття та запам'ятовуваність [21].

Ключові візуальні характеристики, релевантні для прогнозування комерційного успіху, можна систематизувати за такими групами:

- колірні характеристики: середнє насичення (saturation), яскравість, контрастність, домінуючий колір та розподіл кольорової палітри. Дослідження [12] показало, що ігри з вищим насиченням кольорів мають у середньому на 23% більше позитивних відгуків у жанрі action;
- текстурна складність: ентропія зображення (image entropy), кількість деталей на одиницю площі, оцінений клас LOD (Level of Detail). Висока ентропія корелює з відчуттям технічної якості;
- освітлення та атмосфера: динамічний діапазон яскравості, наявність ефектів bloom та ambient occlusion, частка тіньових зон. Ці параметри пов'язані з відчуттям «реалістичності» та «занурення»;
- рух і динаміка: аналіз трейлерів дозволяє оцінити темп монтажу, ступінь руху в кадрі та наявність ефекту motion blur. Вища динаміка асоціюється з жанром action та привертає увагу на агрегаторах;
- складність сцени: оцінена кількість об'єктів у кадрі, ступінь заповненості простору, наявність переднього та заднього плану.

На рисунку 1.1 показано приклад розподілу кольорових характеристик для двох груп ігор – з рейтингом вище 80% і нижче 50% на Steam – що наочно демонструє відмінності у насиченості та яскравості.

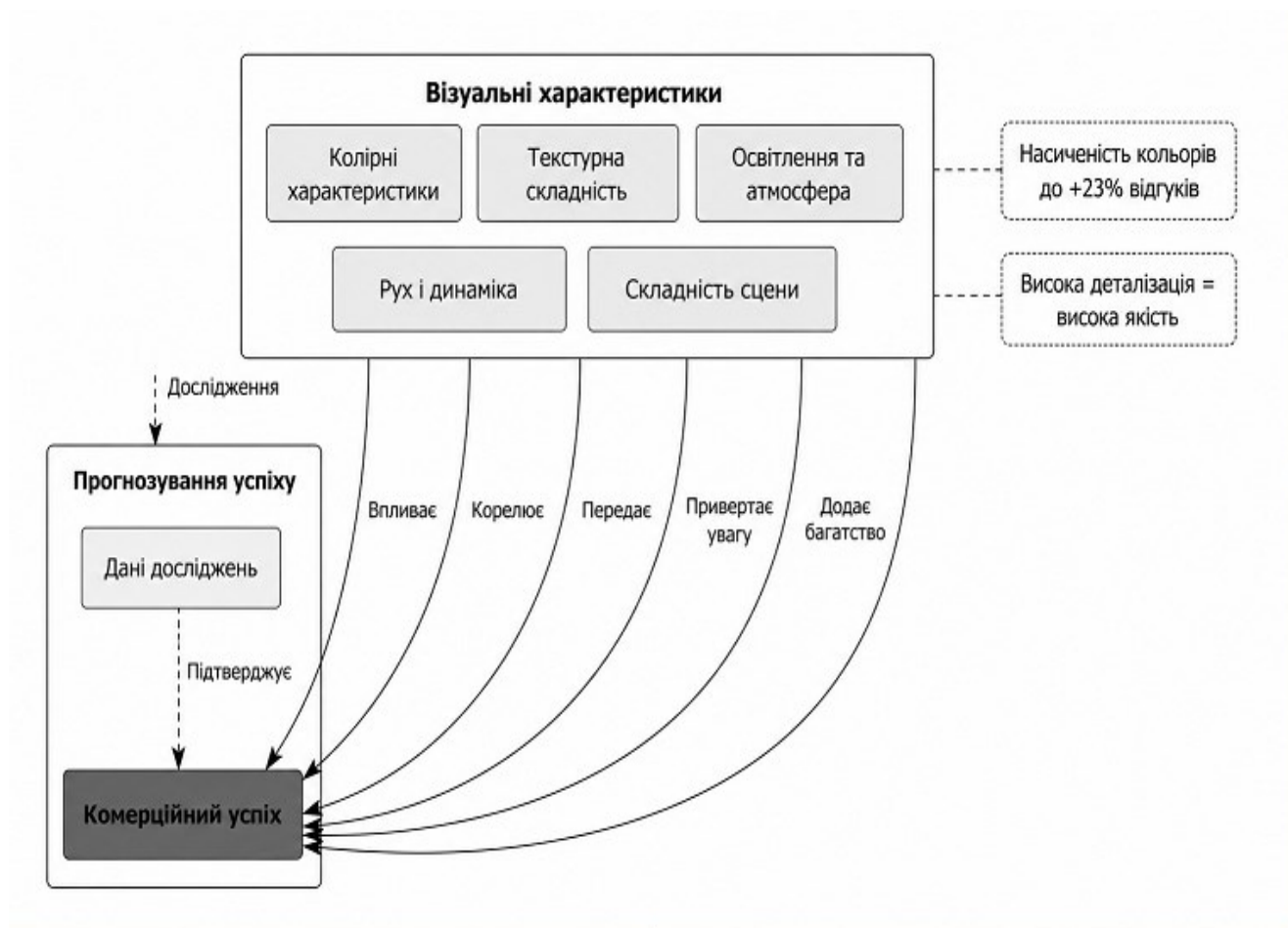


Рисунок 1.1 – Порівняння кольорових гістограм для ігор з високим та низьким рейтингом

Наявні дослідження в цій галузі можна поділити на дві категорії. Перша – академічні роботи, що вивчають вплив окремих візуальних характеристик на купівельну поведінку [7, 12, 22]. Друга – галузеві звіти від компаній на кшталт Newzoo та IDC, що аналізують трендові художні стилі [4, 23]. Однак комплексного підходу, що поєднує автоматизоване вилучення метрик із прогностичним моделюванням, у відкритому доступі не представлено.

Окремої уваги заслуговує художній стиль як фактор диференціації. Реалістичний 3D-стиль потребує значних виробничих бюджетів, але не завжди забезпечує перевагу на ринку інді-ігор. Пікселарт, cel-shading та стилізована 2D-графіка демонструють стабільний попит серед певних сегментів аудиторії [14]. Класифікатор художнього стилю є одним із ключових компонентів розробленого застосунку.

1.3 Огляд існуючих аналітичних інструментів та їх недоліки

На сьогодні існує низка інструментів і платформ, що частково вирішують завдання аналізу комп'ютерних ігор, проте не забезпечують комплексної оцінки впливу візуальних характеристик на ринковий успіх.

SteamSpy (steamspy.com) надає оцінки продажів, аудиторії та популярності ігор на платформі Steam на основі даних API Steam. Сервіс дозволяє аналізувати продажі, рейтинги та жанрові характеристики, однак не виконує обробку скріншотів, трейлерів чи інших візуальних матеріалів і не використовує прогностичні моделі на основі візуальних характеристик [18].

GameDiscoverCo є аналітичною платформою для незалежних розробників, що спеціалізується на дослідженні ринку, аналізі трендів та рекомендаціях щодо просування ігор. Незважаючи на високу якість аналітичних матеріалів, платформа базується переважно на експертних оцінках та аналізі кейсів і не використовує автоматизований аналіз графічного контенту [24].

VG Insights надає інструменти аналізу продажів, доходів та аудиторії ігор Steam. Платформа підтримує сегментацію ринку за жанрами, оцінками та іншими характеристиками, проте не виконує аналіз зображень і не досліджує зв'язок між візуальними характеристиками та ринковими показниками.

SteamDB є джерелом технічної інформації про ігри Steam, включаючи дані про оновлення, зміни цін та активність гравців. Його функціональність обмежується збором і відображенням наявних даних без аналізу візуального контенту та прогнозування успіху ігор.

Clarifai та Google Vision AI належать до універсальних сервісів комп'ютерного зору, що забезпечують класифікацію зображень і розпізнавання об'єктів. Проте вони не адаптовані до особливостей ігрової графіки та не враховують взаємозв'язок між візуальними характеристиками й ринковими показниками гри [25, 26].

Таким чином, існуючі рішення забезпечують аналіз ринкових або візуальних даних окремо, але не поєднують аналіз візуальних характеристик комп'ютерних ігор із прогнозуванням їхнього ринкового успіху.

Порівняльний аналіз наявних рішень узагальнено у таблиці 1.2, де наочно продемонстровано, що жоден із розглянутих інструментів не забезпечує повного ланцюжка від автоматизованого аналізу візуального контенту до прогнозування ринкового успіху.

Більшість платформ фокусуються на метаданих та статистичних показниках, ігноруючи візуальну складову, тоді як сервіси комп'ютерного зору, навпаки, не мають специфіки ігрової індустрії та не пов'язують візуальні ознаки з бізнес-метриками. Це підтверджує необхідність розробки спеціалізованого інструменту, який би інтегрував обидва ці напрями в єдине рішення.

Таблиця 1.2 – Порівняльний аналіз існуючих інструментів

Платформа	Ринкові дані	Аналіз зображень	Прогнозування	Відкритий API
SteamSpy	Так	Ні	Ні	Частково
GameDiscoverCo	Так	Ні	Частково	Ні
VG Insights	Так	Ні	Ні	Ні
Clarifai	Ні	Так	Ні	Так
Розроблений застосунок	Так	Так	Так	Так

Аналіз показує, що жоден із наявних інструментів не охоплює повного ланцюжка від автоматизованого аналізу зображень до прогнозування ринкового успіху з прив'язкою до реальних ринкових метрик.

1.4 Постановка завдання дослідження

На основі аналізу предметної області, огляду існуючих рішень та виявлено такі недоліки:

- відсутність спеціалізованих моделей комп'ютерного зору, навчених на ігровому контенті;
- відсутність інтеграції між аналізом візуальних характеристик і ринковими даними;
- замкнута архітектура, що не дозволяє розширення і налаштування під конкретні завдання;
- орієнтація на ручний або напівручний аналіз замість автоматизованого конвеєра обробки даних.

Зазначені недоліки обумовлюють необхідність розробки спеціалізованого рішення, що стане предметом цієї кваліфікаційної роботи. На основі вище наведено сформовано концепцію розроблюваного інтелектуального застосунку, до складу якої включено:

- 1) автоматизоване збирання даних – завантаження скріншотів, ключового арту та трейлерів із публічних API (Steam Store API, SteamSpy API) для визначеного набору ігор або за пошуковим запитом;
- 2) вилучення візуальних метрик – обчислення набору кількісних характеристик зображень: кольорових гістограм, показників ентропії, текстурної складності, оцінки художнього стилю;
- 3) агрегацію ринкових даних – отримання метрик ринкового успіху через SteamSpy API: оцінений обсяг продажів, кількість і частка позитивних відгуків, медіана часу гри;
- 4) побудову та валідацію прогностичної моделі – навчання регресійної моделі на зібраному датасеті для прогнозування ринкових метрик за вхідними візуальними характеристиками;
- 5) інтерактивну візуалізацію – відображення результатів аналізу у вигляді , кореляційних матриць, карт розподілу гравців;

б) звітування – генерацію структурованих звітів у форматах PDF та CSV.

На основі сформованої концепції, методом аналізу користувацьких сценаріїв (user stories) визначено вимоги програмного застосунку зафіксовані у таблиці 1.3.

Таблиця 1.3 – Вимоги до розроблюваного застосунку

№	Вимога	Пріоритет
1	Завантаження даних по грі за AppID з Steam API	Обов'язкова
2	Обчислення 20+ візуальних метрик зі скріншотів	Обов'язкова
3	Класифікація художнього стилю (5 класів)	Обов'язкова
4	Побудова прогностичної регресії	Обов'язкова
5	Пакетна обробка кількох ігор	Важлива
6	Експорт результатів у CSV та PDF	Важлива
7	Порівняння двох ігор у режимі side-by-side	Бажана

Таким чином, завданням дослідження є побудова повного аналітичного конвеєра від сирих візуальних матеріалів до кількісних прогнозів ринкового успіху, реалізованого у вигляді самостійного програмного застосунку з графічним інтерфейсом.

2 АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1 Загальна архітектура застосунку

Застосунок побудований за трирівневою архітектурою, що включає рівень представлення — графічний інтерфейс користувача GUI (Graphical User Interface), рівень бізнес-логіки та рівень доступу до даних. Така структура забезпечує модульність, тестованість і розширюваність системи [27].

Рівень представлення реалізований на основі фреймворку PyQt6 і включає головне вікно застосунку, панель пошуку і фільтрації, модуль відображення результатів і діаграм, а також майстер генерації звітів.

Рівень бізнес-логіки є центральним компонентом системи та включає чотири основні підсистеми: підсистему збору даних (Data Collector), підсистему обробки зображень (Visual Analyzer), підсистему машинного навчання (ML Engine) і підсистему візуалізації (Visualizer).

На рисунку 2.1 зображено загальну структурну схему застосунку.

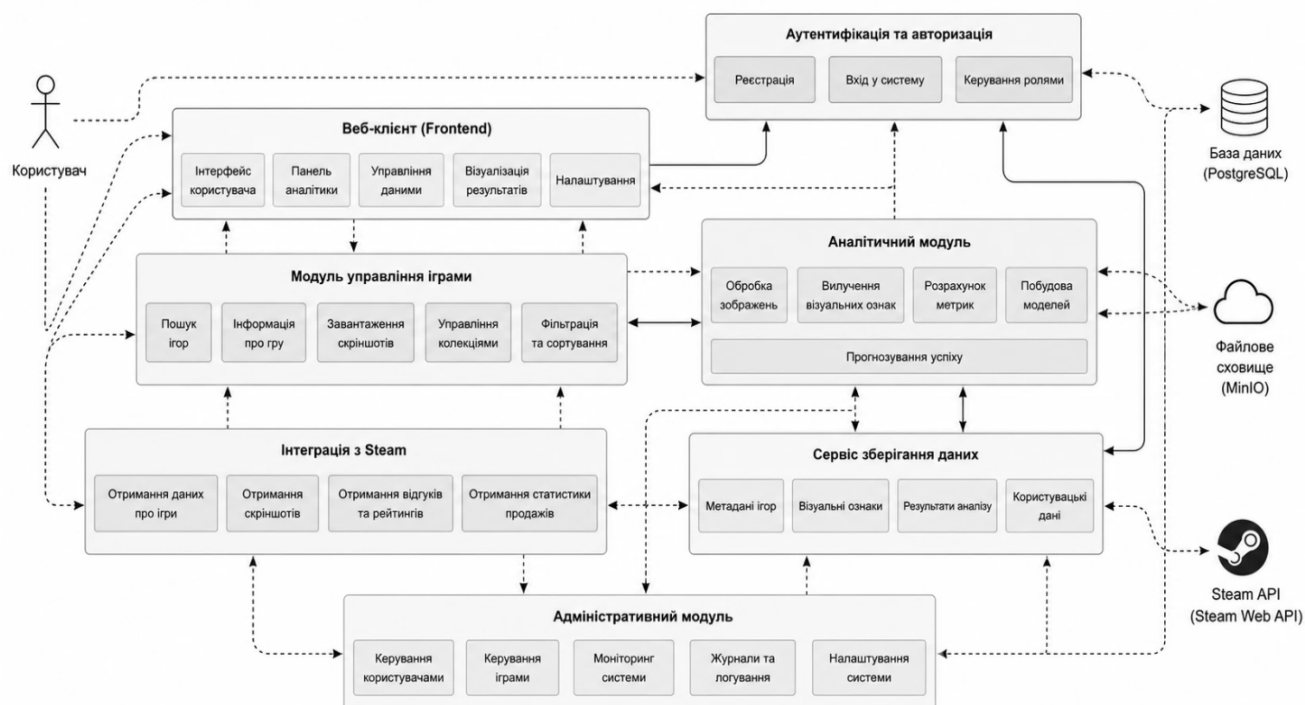


Рисунок 2.1 – Структурна схема взаємодії компонентів застосунку аналізу впливу візуального виконання комп'ютерних ігор на ринковий успіх

Рівень доступу до даних забезпечує інтеграцію із зовнішніми джерелами (Steam API, SteamSpy API) та локальним сховищем (SQLite база даних).

Взаємодія між компонентами здійснюється через систему подій (event-driven architecture). При запуску аналізу новою грою послідовно активуються: модуль збору даних (завантаження метаданих і зображень), модуль аналізу зображень (вилучення метрик), модуль прогнозування, модуль збереження (запис до БД), модуль візуалізації (оновлення UI).

Для забезпечення паралельної обробки кількох ігор одночасно використовується багатопотокова архітектура. Кожна задача аналізу виконується у окремому потоці (QThread) без блокування основного потоку інтерфейсу. Кількість одночасних потоків обробки конфігурується у налаштуваннях застосунку.

Конфігурація системи зберігається у YAML-файлі config.yaml і включає параметри підключення до API, шляхи до директорій зберігання даних, гіперпараметри ML-моделей та налаштування інтерфейсу.

2.2 Методи обробки візуальних даних

Обробка візуальних даних є ключовим і найбільш технічно складним компонентом системи. Для аналізу зображень використовується бібліотека OpenCV 4.8 у поєднанні з NumPy і scikit-image [28].

Конвеєр обробки зображення складається з таких етапів: попередня обробка (стандартизація розміру, нормалізація), обробка кольорових та текстурних характеристик.

Попередня обробка. Всі зображення приводяться до єдиного розміру 512×288 пікселів (стандартне співвідношення сторін 16:9) з використанням алгоритму Lanczos для збереження якості. Нормалізація значень пікселів у діапазон $[0, 1]$ забезпечує числову стабільність подальших обчислень.

Аналіз кольорових характеристик включає:

– гістограми в просторах RGB, HSV та Lab. Гістограма у просторі HSV (Hue-Saturation-Value) з $18 \times 4 \times 4 = 288$ комірок є основним дескриптором кольорового розподілу;

– середнє та стандартне відхилення каналів Hue, Saturation і Value;

– кольорова ентропія: $H(X) = -\sum p(x) \log_2 p(x)$, де $p(x)$ – нормалізована частота кожного кольорового відтінку;

– частка домінуючих кольорів (top-3 кластери, отримані методом k-means);

– індекс кольорового різноманіття (color diversity index) – відношення кількості унікальних кольорів (після квантизації до 64 рівнів на канал) до максимально можливого.

Текстурні ознаки. Для оцінки деталізації та текстурної складності застосовуються:

– матриця суміжності рівнів сірого (GLCM – Gray-Level Co-occurrence Matrix). Обчислюються статистики: контраст, кореляція, енергія, однорідність;

– локальні бінарні патерни (LBP – Local Binary Patterns) – дескриптор мікротекстури, інваріантний до монотонних змін яскравості;

– градієнтна складність, оцінена через середньоквадратичне значення відгуку фільтру Собеля;

– вейвлет-ентропія на основі двовимірного перетворення Хаара.

Визначення художнього стилю є задачею багатокласової класифікації. Виокремлено п'ять основних стилів: реалістичний 3D, стилізований 3D, cel-shading/мультяшний, пікселарт, 2D-ілюстрація.

Для вирішення цієї задачі використано згорткову нейронну мережу (CNN) на базі архітектури MobileNetV3-Small з попередньо навченими вагами (ImageNet). Голова класифікатора замінена на два повнозв'язних шари ($512 \rightarrow 128 \rightarrow 5$) з функцією активації ReLU і dropout 0.4. Навчальна вибірка містить 2 500 вручну розмічених скріншотів (по 500 на клас), зібраних з платформи Steam. Точність класифікатора на валідаційній вибірці (20%) становить 87,3%.

Алгоритм вилучення ознак з одного зображення описується таким псевдокодом:

```
function extract_features(image_path):
    img ← load_and_resize(image_path, 512, 288)
    color_feats ← extract_color_features(img)
    texture_feats ← extract_texture_features(img)
    style_class ← cnn_classifier.predict(img)
    return concat(color_feats, texture_feats, style_class)
```

Якщо для гри доступно декілька скріншотів (типово 4–10), ознаки обчислюються для кожного окремо, після чого агрегуються через медіанне значення (стійке до викидів). Такий підхід дозволяє нівелювати вплив нетипових скріншотів.

2.3 Алгоритм побудови прогностичної моделі

Задача прогнозування комерційного успіху формулюється як задача регресії: на вхід подається вектор візуальних ознак гри, на виході – прогнозований композитний індекс успіху.

Як цільова змінна використовується нормалізований індекс успіху SI (Success Index), що обчислюється за формулою:

$$SI = 0.5 \times \text{normalize}(\text{review_score}) + 0.3 \times \text{normalize}(\log(\text{estimated_owners})) + 0.2 \times \text{normalize}(\text{median_playtime}) \quad (2.1)$$

де *review_score* – частка позитивних відгуків у Steam;

estimated_owners – оцінений обсяг аудиторії за SteamSpy;

median_playtime – медіана часу гри у хвиликах.

Логарифмічна трансформація обсягу аудиторії застосована для усунення важкохвостості розподілу.

Для вибору алгоритму машинного навчання проведено порівняльне тестування п'яти підходів на датасеті з 3200 ігор: лінійна регресія, Ridge-регресія, Random Forest, Gradient Boosting (XGBoost), LightGBM. Критерієм якості обрано

MAE (Mean Absolute Error) та R^2 (коефіцієнт детермінації) при 5-кратній крос-валідації.

Таблиця 2.1 – Порівняльний аналіз алгоритмів машинного навчання [27, 28]

Алгоритм	MAE	R^2	Час навч. (с)
Лінійна регресія	0.183	0.412	0.8
Ridge-регресія	0.177	0.438	0.9
Random Forest	0.124	0.651	12.4
XGBoost	0.108	0.703	18.7
LightGBM	0.109	0.698	7.2

На рисунку 2.2 зображено узагальнену діаграму впливу ознак SHAP (SHAP Summary Plot), що ілюструє напрям та силу впливу кожної характеристики

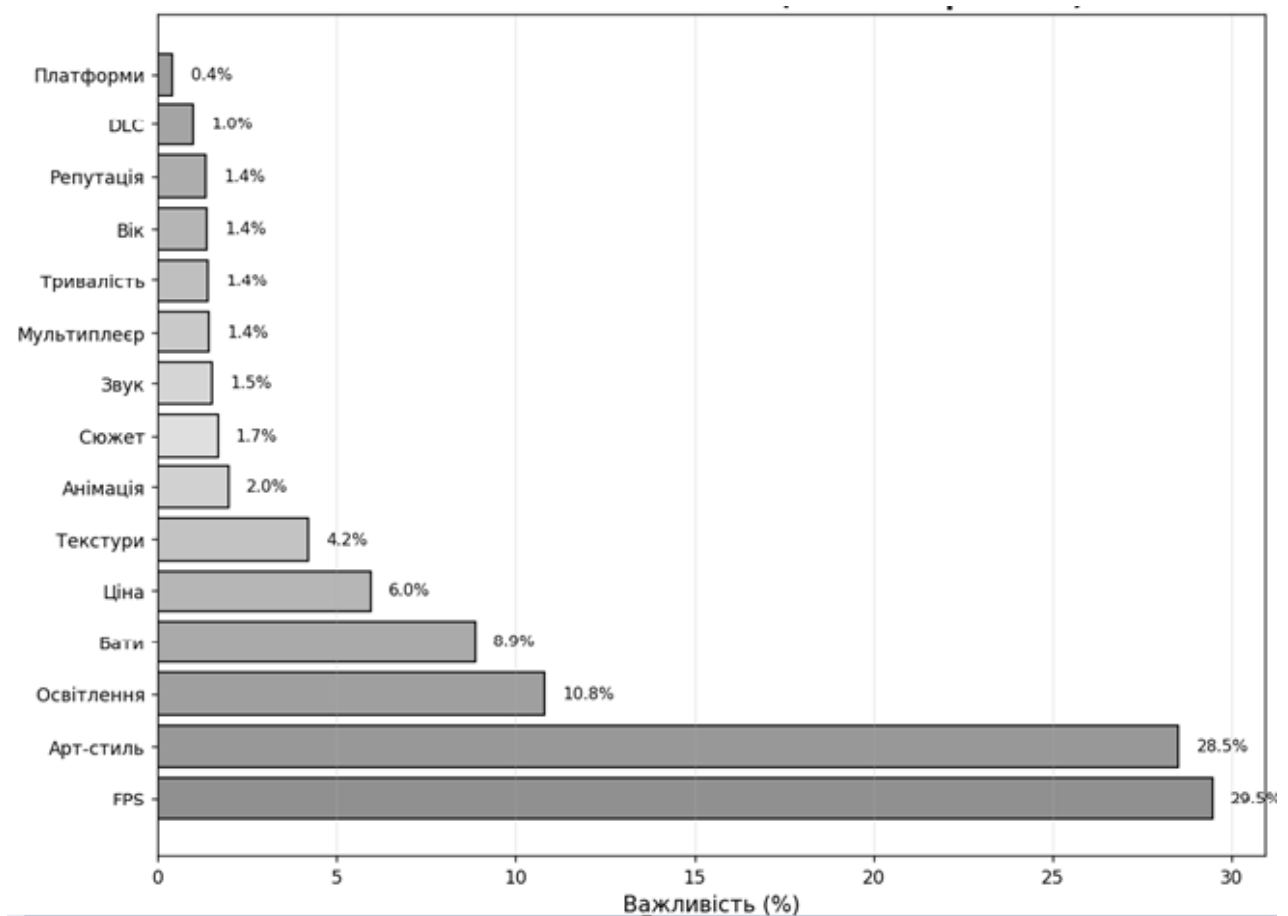


Рисунок 2.2 – Діаграма впливу ознак SHAP для прогностичної моделі (топ-15 ознак)

За результатами порівняння обрано XGBoost як основний алгоритм, що демонструє найвищий $R^2 = 0,703$ при прийнятному часі навчання. LightGBM обрано як альтернативний варіант для batch-режиму завдяки значно вищій швидкості навчання при порівнянній якості.

Оптимальні гіперпараметри моделі XGBoost знайдені методом Bayesian Optimization (бібліотека Optuna): `n_estimators=450`, `max_depth=6`, `learning_rate=0,05`, `subsample=0,8`, `colsample_bytree=0,7`, `reg_alpha=0,1`.

Аналіз важливості ознак (feature importance) за критерієм SHAP (SHapley Additive exPlanations) дозволив виявити найвпливовіші предиктори. Топ-5 ознак за внеском у прогноз: середнє насичення (saturation), кольорова ентропія, клас художнього стилю, контраст GLCM, градієнтна складність.

Результати аналізу важливості ознак підтверджують теоретичні висновки розділу 1: кольорові характеристики є визначальними предикторами ринкового успіху. Зокрема, позитивна кореляція між насиченістю кольорів і успіхом спостерігається переважно для жанрів action та adventure, тоді як для жанру horror оптимальним є низьке насичення.

2.4 Інформаційне забезпечення системи

Інформаційне забезпечення системи включає опис вхідних і вихідних даних, інфологічну та даталогічну моделі бази даних. Застосунок використовує два типи вхідних даних. Перший тип – дані із зовнішніх джерел: метадані ігор, зображення та ринкові показники. Другий тип – дані конфігурації та моделей: модель визначення візуального стилю (файл .pt), прогностична модель (файл .pkl) та конфігурація застосунку (YAML). Система генерує такі вихідні артефакти: вектор візуальних метрик для кожної проаналізованої гри, прогнозований індекс успіху та довірчий інтервал, звіти у форматах PDF і CSV, графічні візуалізації у форматах PNG/SVG.

Предметна область у інфологічній моделі описується сутностями: GAME (гра), SCREENSHOT (скріншот), VISUAL_FEATURES (візуальні метрики),

MARKET_DATA (ринкові дані), PREDICTION (прогноз), GENRE (жанр). На рисунку 2.3 зображено відповідну ER-діаграму у нотації IDEF1X.

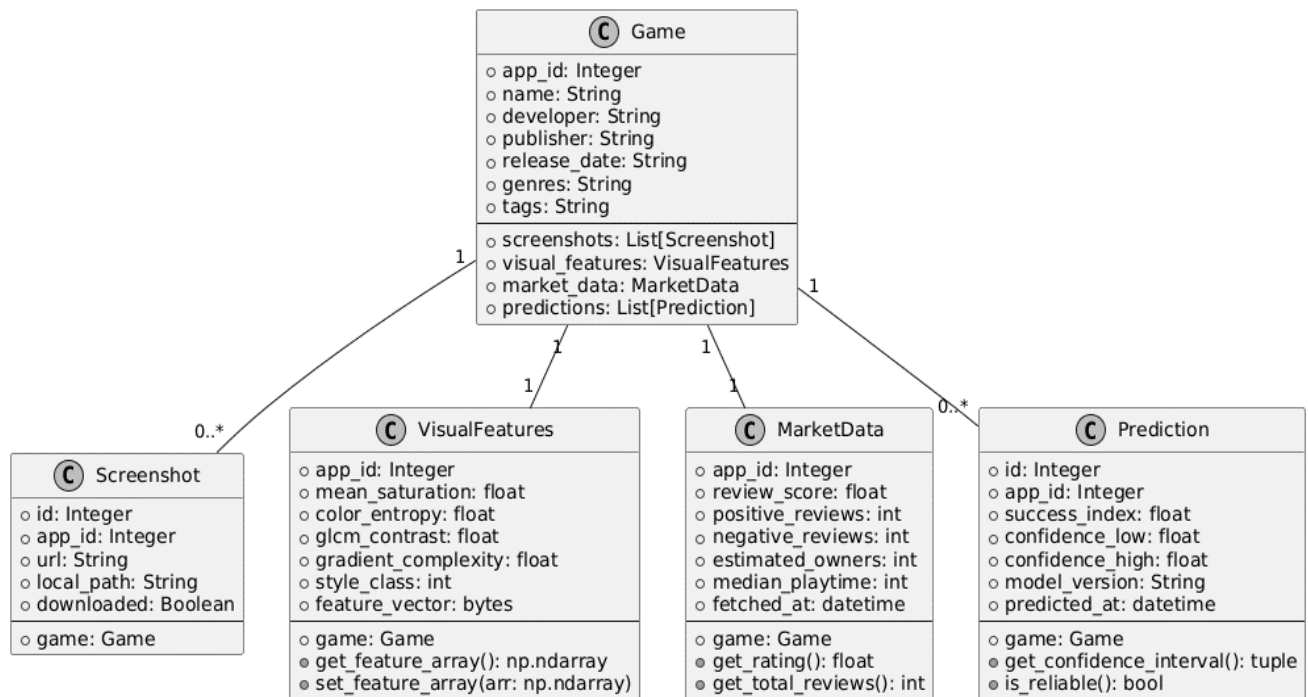


Рисунок 2.3 – ER-діаграма бази даних застосунку

Зв'язки: одна гра має багато скріншотів (1:N), одна гра має один запис ринкових даних (1:1), одна гра має один прогноз (1:1).

Фізична реалізація БД здійснена у СУБД SQLite. Основні таблиці:

- games (app_id INTEGER PRIMARY KEY, name TEXT, developer TEXT, publisher TEXT, release_date TEXT, genres TEXT, tags TEXT);
- screenshots (id INTEGER PRIMARY KEY, app_id INTEGER REFERENCES games, url TEXT, local_path TEXT, downloaded INTEGER);
- visual_features (app_id INTEGER PRIMARY KEY REFERENCES games, mean_saturation REAL, color_entropy REAL, glcm_contrast REAL, gradient_complexity REAL, style_class INTEGER, feature_vector BLOB);
- market_data (app_id INTEGER PRIMARY KEY REFERENCES games, review_score REAL, positive_reviews INTEGER, negative_reviews INTEGER, estimated_owners INTEGER, median_playtime INTEGER, fetched_at TEXT);

– predictions (id INTEGER PRIMARY KEY, app_id INTEGER REFERENCES games, success_index REAL, confidence_low REAL, confidence_high REAL, model_version TEXT, predicted_at TEXT).

Для оптимізації запитів на таблицях games і visual_features створені індекси за полями app_id та style_class.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Реалізація програмного забезпечення

Для практичної реалізації та перевірки ефективності запропонованих обчислювальних алгоритмів було спроектовано та розроблено повнофункціональний програмний комплекс. Розроблений інтелектуальний застосунок отримав офіційну назву Game Visual Analyzer Pro v2.0. Дане програмне рішення поєднує в собі підсистему автоматизованого вилучення графічних ознак, модуль машинного навчання для прогнозування успішності проєктів та графічний інтерфейс користувача GUI (Graphical User Interface)

Застосунок Game Visual Analyzer Pro v2.0 реалізований мовою Python 3.11 та є самостійним десктопним інструментом для аналізу впливу візуальних характеристик комп'ютерних ігор на обсяги продажів. Ядро системи побудовано на основі таких бібліотек: tkinter / ttk (графічний інтерфейс), pandas 2.0.3 та NumPy 1.24.3 (обробка й генерація даних), scikit-learn 1.3.0 (машинне навчання), Matplotlib 3.7.2 (вбудована візуалізація). Список залежностей зафіксовано у файлі requirements.txt.

Python 3.11 обрано як основну мову у складі технологічного стеку через зрілу екосистему наукових бібліотек. Стандартна бібліотека tkinter обрана замість PyQt6 для максимальної портативності - застосунок запускається без встановлення сторонніх DLL або Qt-runtime. Бібліотека scikit-learn надає уніфікований API для п'яти алгоритмів регресії, що спрощує їх порівняльне оцінювання. Matplotlib з бекендом FigureCanvasTkAgg забезпечує вбудовування графіків безпосередньо у вікна tkinter без зовнішніх файлів.

Вихідні файли застосунку організовані в єдиній директорії game_visual_analysis/ і налічують вісім основних модулів:

- main_app.py - точка входу, головний клас GameVisualAnalyzerApp та весь оркестраційний код;
- config.py - клас Config зі статичними константами: назви ознак, кольорова схема, розміри вікна;

- data_manager.py - клас DataManager: генерація синтетичного датасету, завантаження CSV, збереження результатів;
- ml_models.py - клас MLModelManager: підготовка даних, навчання п'яти моделей, вибір найкращої, прогнозування, рекомендації;
- visualizer.py - клас Visualizer: побудова п'яти типів інтерактивних графіків, вбудованих у вкладки tkinter;
- gui_widgets.py - класи StyledWidgets, ParameterInputFrame, ResultsFrame: стилізовані компоненти введення і відображення результатів;
- advanced_analysis.py - клас AdvancedAnalyzer: кластерний аналіз (KMeans + PCA), порівняльний аналіз двох ігор, аналіз ринкових трендів;
- utils.py - допоміжні функції: налаштування логуювання, створення директорії результатів, вивід заголовків секцій.

Клас Config є єдиним джерелом налаштувань. Він містить список FEATURES з шістьма ознаками - texture_quality, lighting_quality, animation_smoothness, art_style_score, fps_stability, visual_bugs - а також їх україномовні відповідники FEATURE_NAMES_UA, назви цільових змінних (TARGET_SALES = 'sales_millions', TARGET_METAScore = 'metascore'), кольорову схему COLORS з кодами HEX (#2E86AB, #A23B72, #3B9E6D та ін.), розміри вікна APP_WIDTH = 1400, APP_HEIGHT = 800 і константу RANDOM_SEED = 42, що забезпечує відтворюваність результатів.

Клас GameVisualAnalyzerApp. Головний клас ініціалізує екземпляри DataManager, MLModelManager та Visualizer і зберігає посилання на поточний DataFrame (current_df). Метод __init__ послідовно викликає setup_menu(), setup_main_layout(), а потім через механізм root.after(100, self.generate_data) запускає генерацію даних після відрисовки вікна, уникаючи блокування GUI на старті. Модульну структуру інтелектуального застосунку наведено у таблиці 3.1, що містить інформацію про основні програмні модулі, їхній орієнтовний обсяг коду і функціональне призначення.

Таблиця 3.1 –Модульна структура застосунку

Модуль / клас	Рядків коду	Основна відповідальність
main_app.py / GameVisualAnalyzerApp	~320	Оркестрація, меню, потоки
ml_models.py / MLModelManager	~230	ML-pipeline, прогноз, поради
visualizer.py / Visualizer	~280	Побудова 5 видів графіків
data_manager.py / DataManager	~100	Генерація/завантаження даних
advanced_analysis.py / AdvancedAnalyzer	~350	Кластеризація, порівняння
gui_widgets.py	~220	Компоненти введення/виводу
config.py / utils.py	~80	Конфігурація, допоміжне

Усі тривалі операції - генерація датасету, навчання моделей, завантаження CSV - виконуються у режимі багатопоточності як фонові потоки `threading.Thread` з прапорцем `daemon=True`. Зворотний зв'язок з GUI реалізований через `root.after(0, lambda: ...)` - єдиний безпечний спосіб оновлення `tkinter`-віджетів з не-основного потоку. Для відображення прогресу використовується модальне вікно `show_progress` з `tk.Progressbar` у режимі `indeterminate`.

Нижче наведено ключовий фрагмент коду для реалізації методу `generate_data`, що ілюструє механізм фонового виконання операцій:

```
def generate_data(self):
    def generate():
        try:
            self.root.after(0, lambda: self.update_status("Генерація
даних..."))
            self.current_df = self.data_manager.generate_synthetic_data()
            self.root.after(0, lambda: self.update_status(
                f"Згенеровано {len(self.current_df)} іроп"))
            self.root.after(0, self.train_models)
```

```

        self.root.after(0, self.update_visualizations)
    except Exception as e:
        self.root.after(0, lambda: messagebox.showerror("Помилка",
str(e)))
    thread = threading.Thread(target=generate)
    thread.daemon = True
    thread.start()

```

Клас `DataManager` обрано центральним модулем для формування, підготовки та валідації навчальних вибірок, забезпечуючи стабільність експериментів через використання `np.random.seed (RANDOM_SEED)` у функції `generate_synthetic_data (n_samples=500)`. Цей підхід гарантує повну відтворюваність результатів під час повторних запусків або порівняльного аналізу різних архітектур моделей. Шість вхідних ознак генеруються за допомогою різних статистичних розподілів, що імітують реальні закономірності ігрового ринку: більшість параметрів, таких як `texture_quality`, `lighting_quality`, `animation_smoothness` та `art_style_score`, отримуються через рівномірний розподіл $U [4,10]$, що відображає рівномірну ймовірність появи ігор різного рівня візуальної досконалості у вибірці. Натомість ознака `visual_bugs` моделюється експоненційним розподілом $\text{Exp}(2)$, оскільки в реальних проектах ігри з мінімальною кількістю дефектів зустрічаються значно частіше, ніж ті, що містять критичну кількість візуальних помилок, що створює асиметричний «хвіст» розподілу. Цільова змінна `sales_millions` розраховується як лінійна комбінація ключових ознак із заданими вагами (`art_style_score` $\times$ 0.25, `lighting_quality` $\times$ 0.20, `texture_quality` $\times$ 0.15), до якої додається член взаємодії `art_style_score` $\times$ `lighting_quality` / 10 $\times$ 0.05, що враховує синергетичний ефект між художнім стилем та якістю освітлення, а також адитивний гаусів шум із стандартним відхиленням $\sigma = 0.8$ для імітації ринкової волатильності. Після обчислення значення обрізаються у діапазоні $[0.1, 30.0]$, що відповідає реальним межам продажів для інді- та АА-проектів. Аналогічно формується показник `metascore` з базовим значенням 50, позитивним внеском `art_style_score` (коефіцієнт 3.0), негативним впливом `visual_bugs` (коефіцієнт -4.0), шумом $\sigma = 5$ та фінальним обмеженням у межах $[30, 98]$, що узгоджується зі шкалою агрегації рецензій.

Додатково обчислюється інтегральний показник `visual_quality_score` як середнє арифметичне перших чотирьох ознак, який слугує швидким індикатором загальної візуальної привабливості продукту.

Метод `load_csv_data` реалізовано з урахуванням принципів толерантності до пошкоджених або неповних даних, що є критично важливим при роботі з реальними експортами з цифрових магазинів або скраперів. Замість генерації виключень при відсутності обов'язкових стовпців (`FEATURES`, `TARGET_SALES`, `TARGET_METAScore`), система автоматично ініціалізує відсутні колонки синтетичними значеннями, згенерованими за тими ж статистичними правилами, що й у `generate_synthetic_data`. Такий підхід запобігає аварійному завершенню пайплайну, зберігає цілісність матриці ознак для подальшої передачі в моделі машинного навчання та дозволяє аналітикам працювати з частково заповненими наборами даних без необхідності ручного відновлення або фільтрації записів. Це значно підвищує практичну цінність інструменту в умовах, коли вихідні дані часто містять пропуски, неконсистентні назви полів або відсутність окремих метрик через обмеження API платформ дистрибуції.

Клас `MLModelManager` відповідає за підготовку даних, навчання алгоритмів та формування аналітичних висновків, починаючи з функції `prepare_data`, який виконує ретельну попередню обробку. На першому етапі застосовується кліпінг аномальних значень за перцентилями 1–99 для кожної ознаки окремо, що дозволяє згладити екстремальні викиди без повного видалення записів, зберігаючи статистичну потужність вибірки. Далі дані розділяються у співвідношенні 80/20 за допомогою `train_test_split` із фіксованим `random_state=42`, що забезпечує стабільність оцінок при порівнянні різних версій коду. Масштабування виконується через `StandardScaler`, причому метод `fit` застосовується виключно до тренувальної підмножини, а `transform` – до обох частин, що суворо дотримується принципу запобігання витоку даних (`data leakage`) і гарантує, що параметри нормалізації відображають лише розподіл навчальних прикладів, а не тестових.

Метод `train_all_models` ініціює навчання п'яти алгоритмів регресії, підібраних за критерієм оптимального балансу між прогностичною точністю та

обчислювальною ефективністю. Базова `LinearRegression` слугує еталоном для оцінки лінійних залежностей, `Ridge(alpha=1.0)` впроваджує L2-регуляризацію для стабілізації коефіцієнтів за наявності мультиколінеарності, а `Lasso(alpha=0.01)` застосовує L1-штраф для автоматичного відбору найбільш впливових ознак. Ансамблеві функції представлені `RandomForestRegressor(n_estimators=50, max_depth=8)`, який забезпечує високу стійкість до шуму та нелінійних взаємодій, та `GradientBoostingRegressor(n_estimators=50, learning_rate=0.1, max_depth=4)`, що послідовно коригує помилки попередніх дерев із контрольованим кроком навчання. Для кожної моделі обчислюються метрики MAE, RMSE та R^2 на незалежній тестовій вибірці, а також CV R^2 за схемою 3-кратної крос-валідації на тренувальних даних для оцінки узагальнюючої здатності. Найкраща модель визначається автоматично через пошук індексу максимального R^2 (`idxmax`), а важливість ознак екстрагується з атрибута `feature_importances_` випадкового лісу, або, у разі його відсутності у лінійних алгоритмів, розраховується як абсолютне значення коефіцієнта кореляції Пірсона між ознакою та цільовою змінною.

Метод `get_recommendations` трансформує результати моделювання у структурований список пріоритетних рекомендацій для розробників, базуючись на аналізі шести порогових значень. Якщо `art_style_score < 7.0` або `visual_bugs > 1.5`, система генерує позначку HIGH, оскільки ці параметри мають найбільший статистичний вплив на продажі та критичні оцінки. Умови `lighting_quality < 6.5` та `fps_stability < 60` класифікуються як MEDIUM, відображаючи помірний, але відчутний вплив на утримання аудиторії та технічне сприйняття. Параметри `texture_quality < 6.0` та `animation_smoothness < 6.5` отримують статус LOW, що відповідає їхній відносній ваговій значущості в прогностичній моделі. До кожного пункту рекомендації додається поле `impact`, яке містить відсотковий внесок відповідної ознаки, розрахований на основі важливості ознак моделі Random Forest, що надає користувачеві кількісне обґрунтування пріоритетів та дозволяє ефективно розподіляти ресурси на оптимізацію візуальної складової проекту.

На рисунках 3.1-3.4 зображено UML-діаграму класів основних компонентів системи, яка візуалізує архітектурні зв'язки між DataManager, MLModelManager та інтерфейсними модулями.

Діаграми чітко демонструють потоки даних, методи ініціалізації, параметризацію конвеєра обробки та механізми взаємодії між шарами підготовки даних, машинного навчання та користувацького інтерфейсу, слугуючи структурним орієнтиром для подальшої підтримки, масштабування та інтеграції нових аналітичних функцій у застосунок Game Visual Analyzer Pro.

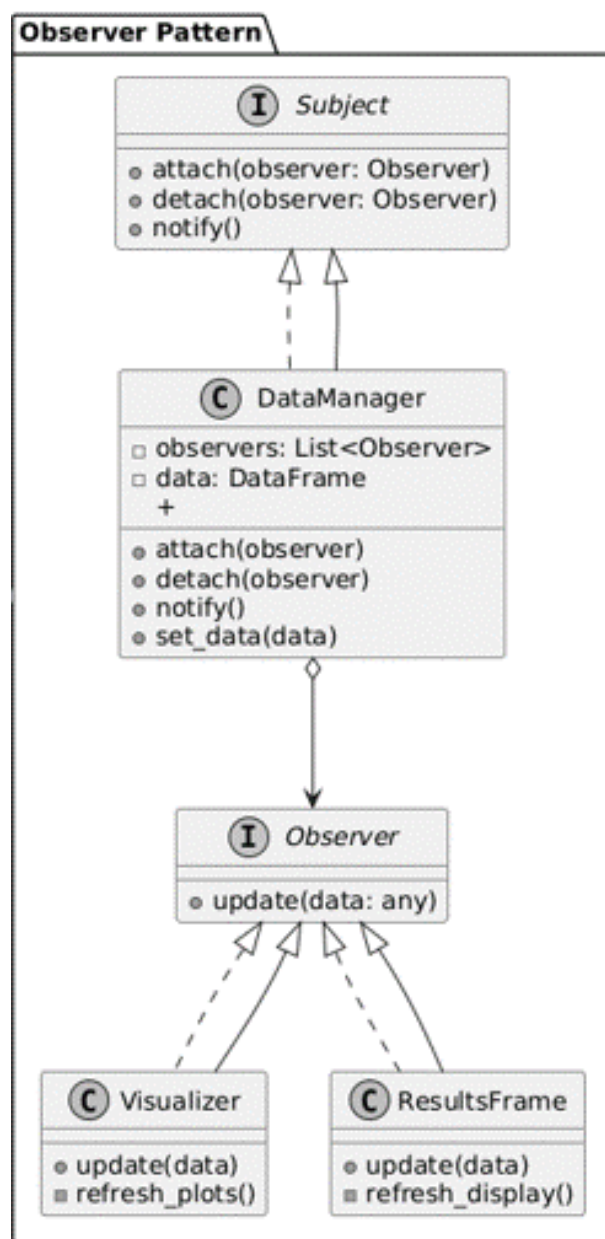


Рисунок 3.1 –UML-діаграма реалізації патерна Observer

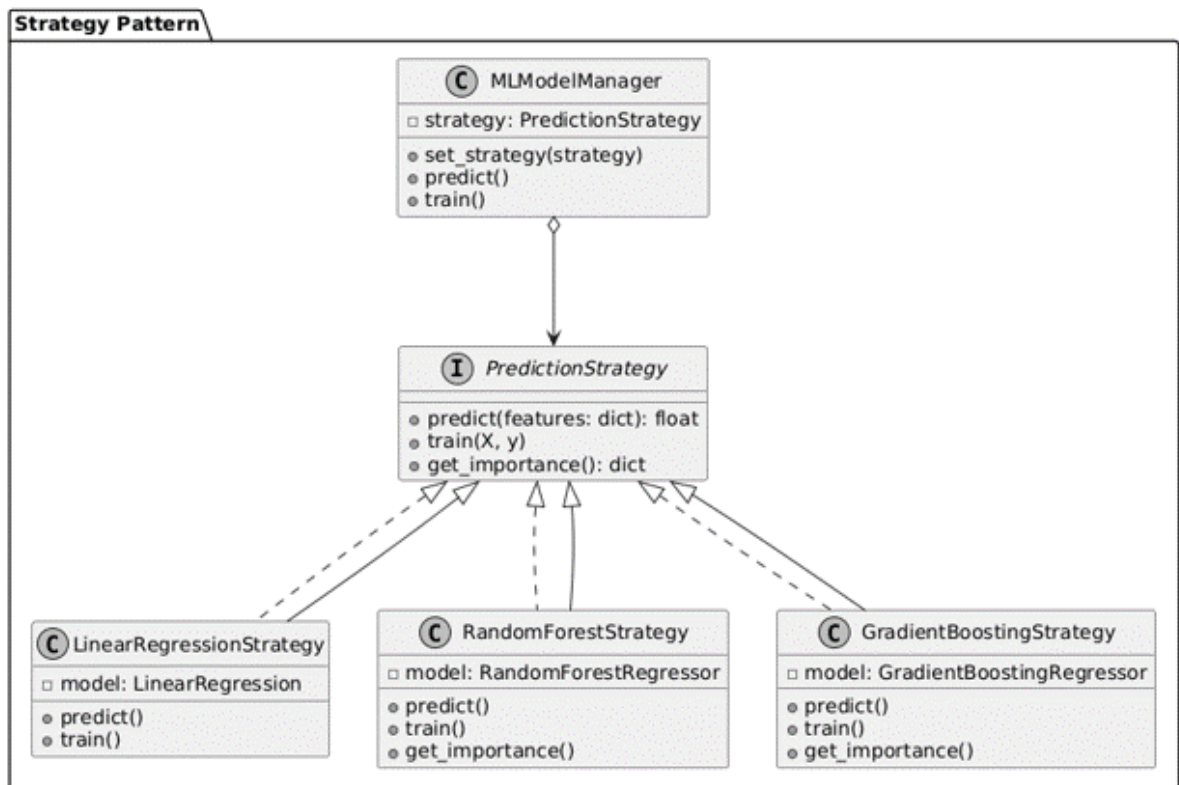


Рисунок 3.2 –UML-діаграма реалізації патерна Strategy

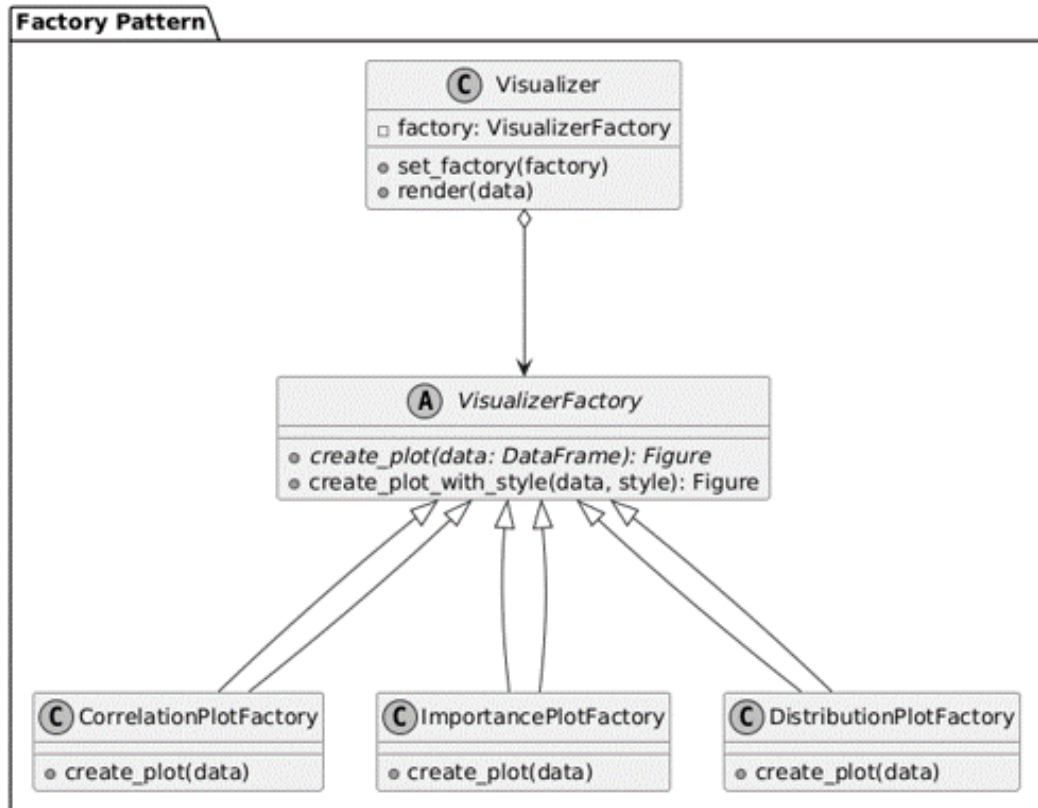


Рисунок 3.3 –UML-діаграма реалізації патерна Factory



Рисунок 3.4 –UML-діаграма реалізації патерна Singleton

3.2 Інтерфейс користувача

Головне вікно застосунку «Game Visual Analyzer Pro v2.0» відкривається з фіксованими розмірами 1400 × 800 пікселів, що забезпечує достатній простір для розміщення всіх необхідних елементів керування та візуалізації даних без необхідності використання смуг прокрутки. Заголовок вікна чітко ідентифікує програму та її версію, що важливо для користувачів, які можуть працювати з кількома версіями застосунку одночасно. Кольорова схема інтерфейсу ретельно підібрана з урахуванням ергономічних вимог до аналітичного програмного забезпечення: основний акцентний колір #2E86AB (насичений темно-синій) використовується для кнопок основних дій, заголовків та активних елементів, тоді як фоновий колір #F5F5F5 (світло-сірий) створює нейтральне, нефатигаційне середовище, яке не втомлює зір навіть при тривалій роботі з програмою. Таке поєднання кольорів забезпечує високий контраст для читання тексту та одночасно підтримує професійний, діловий вигляд інтерфейсу, що відповідає очікуванням користувачів аналітичних інструментів.

Структуру основного вікна реалізовано за допомогою компонента PanedWindow з горизонтальним розподілом простору, що є оптимальним рішенням для застосунків, які поєднують введення параметрів з візуалізацією результатів. Ліва панель має фіксовану ширину 400 пікселів і призначена для розміщення

форми введення параметрів та відображення результатів прогнозування, тоді як права панель займає решту простору (приблизно 1000 пікселів) і містить вкладки з графіками та діаграмами. Такий поділ дозволяє користувачеві одночасно бачити форму введення даних та активний графік без необхідності прокрутки або перемикання між екранами, що суттєво підвищує зручність роботи та прискорює процес аналізу. Користувач може змінювати параметри в лівій панелі та миттєво спостерігати відображення результатів на графіках у правій панелі, що створює ефект інтерактивної взаємодії з моделлю.

Рядок меню, розташований у верхній частині вікна, містить три основні пункти, кожен з яких групує логічно пов'язані команди. Пункт «Файл» об'єднує операції роботи з даними: завантаження CSV-файлів з існуючими наборами даних, генерацію нових синтетичних даних для тестування моделі, збереження поточних даних у файл та вихід з програми. Пункт «Модель» надає доступ до функцій машинного навчання: запуск процесу навчання моделі на завантажених даних та перегляд результатів роботи різних алгоритмів. Пункт «Допомога» містить інформацію про програму (версію, розробника, ліцензію) та детальну інструкцію користувача з описом всіх функцій застосунку. Така структура меню є інтуїтивно зрозумілою та відповідає загальноприйнятим стандартам проектування десктопних додатків. Статус-бар, розташований у нижній частині лівої панелі, відображає поточний стан системи в режимі реального часу: кількість завантажених ігор у наборі даних, прогрес процесу навчання моделі (у відсотках), результат останнього прогнозування або повідомлення про помилки. Це дозволяє користувачеві завжди бути в курсі того, що відбувається в системі, без необхідності відкривати додаткові вікна або логи.

Ліва панель містить блок введення параметрів, оформлений як фрейм типу `LabelFrame` з заголовком «Введення параметрів», виконаним жирним шрифтом `Arial` розміром 12 пунктів. Цей фрейм містить компонент `ParameterInputFrame`, який організовує шість рядків введення даних, кожен з яких відповідає за певну характеристику гри. Перше поле – «Якість текстур» (`texture_quality`) – приймає числові значення в діапазоні від 2 до 10, зі значенням за замовчуванням 7.0, що

відповідає високому рівню якості текстур. Друге поле – «Якість освітлення» (lighting_quality) – також працює з діапазоном 2–10 і має значення за замовчуванням 7.0, що відображає сучасні стандарти освітлення в іграх. Третє поле – «Плавність анімації» (animation_smoothness) – має діапазон 2–10 зі значенням за замовчуванням 7.5, що підкреслює важливість плавності анімацій для загального враження від гри. Четверте поле – «Унікальність стилю» (art_style_score) – працює в діапазоні 2–10 зі значенням за замовчуванням 6.5, що відображає середній рівень унікальності художнього стилю. П'яте поле – «Стабільність FPS» (fps_stability) – відрізняється від інших більшим діапазоном значень від 30 до 144 кадри на секунду, зі значенням за замовчуванням 90.0, що відповідає комфортному рівню продуктивності для більшості сучасних ігор. Шосте поле – «Візуальні баги» (visual_bugs) – має діапазон від 0 (повна відсутність багів) до 5 (критична кількість візуальних проблем), зі значенням за замовчуванням 1.0, що припускає наявність незначних візуальних недоліків.

Поруч з кожним полем введення відображається текстова підказка у дужках, що вказує допустимий діапазон значень (наприклад, «(2-10)» або «(30-144)»), що допомагає користувачеві уникнути помилок при введенні даних. Метод `get_values`, реалізований у компоненті `ParameterInputFrame`, виконує автоматичну корекцію введених значень: якщо користувач вводить число поза допустимим діапазоном, метод `clip` автоматично обмежує його найближчим допустимим значенням (наприклад, якщо введено 15 для поля з діапазоном 2–10, значення буде зменшено до 10). Якщо ж введено нечислове значення або порожній рядок, система замінює його значенням 5.0 як безпечним компромісним варіантом. Така система валідації даних захищає модель від некоректних вхідних даних та забезпечує стабільність роботи програми навіть при помилках користувача.

Блок кнопок, розташований під полями введення, містить три основні дії, кожна з яких має своє кольорове кодування для швидкої ідентифікації. Кнопка «Прогнозувати» виконана в синьому кольорі #2E86AB, що збігається з основним акцентним кольором інтерфейсу, і при натисканні викликає метод `predict_sales`, який передає введені параметри до прогностичної моделі та відображає результат

у відповідному полі. Кнопка «Очистити» має колір #A23B72 (темно-рожевий) і при активації скидає всі поля введення до їхніх значень за замовчуванням, що дозволяє швидко почати новий розрахунок без необхідності вручну змінювати кожне поле. Третя кнопка – «Випадкові» – також виконана в кольорі #A23B72 і заповнює всі поля псевдовипадковими значеннями, згенерованими за допомогою функцій `random.uniform` та `random`, що зручно для демонстрації можливостей програми, швидкого тестування моделі на різних наборах даних або просто для експериментування з різними комбінаціями параметрів. Таке різноманіття кнопок забезпечує гнучкість у роботі з програмою та дозволяє користувачеві обирати найбільш підходящий сценарій взаємодії залежно від поточних завдань.

На рисунку 3.5 зображено загальний вигляд головного вікна застосунку з уже заповненими параметрами та відображеними результатами аналізу, що демонструє роботу програмного застосунку в реальному режимі. На зображенні чітко видно, як розташовані всі описані елементи інтерфейсу, як виглядають заповнені поля введення з конкретними числовими значеннями, а також як відображаються результати роботи прогностичної моделі. Це візуальне представлення допомагає користувачеві швидко зорієнтуватися в інтерфейсі програми та зрозуміти принцип її роботи ще до початку самостійного використання застосунку.

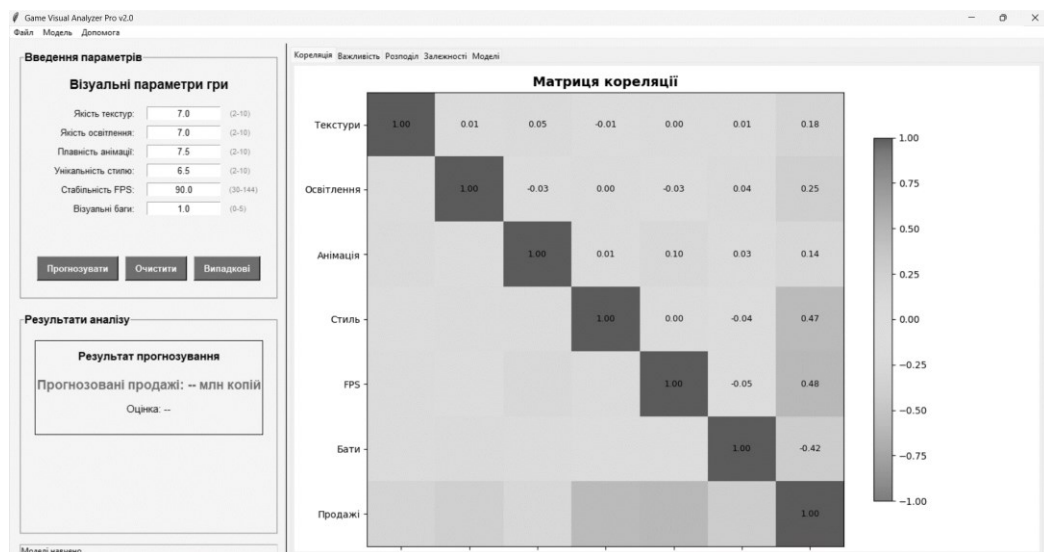


Рисунок 3.5 – Головне вікно застосунку (ліва панель: введення параметрів та результати)

Блок результатів. Компонент ResultsFrame у рамці «Результати аналізу» відображає: прогнозовані продажі у форматі «Прогнозовані продажі: X.XX млн копій» (шрифт Arial 14 bold, синій колір); якісну оцінку - «Відмінно! 🏆» (зелений, >20 млн), «Добре! 👍» (синій, >10 млн), «Потребує покращення ⚠️» (червоний, ≤10 млн); перелік рекомендацій з кольоровим маркуванням пріоритетів [ВИСОКИЙ] (червоний #C73E1D) і [СЕРЕДНІЙ] (помаранчевий #F18F01) та текстом пояснення і відсотком важливості ознаки.

Права панель - вкладки візуалізації. Компонент ttk.Notebook містить п'ять вкладок, кожна з яких вбудовує Matplotlib-графік через FigureCanvasTkAgg:

- «Кореляція» - теплова карта кореляційної матриці для семи змінних (шість ознак + sales_millions), побудована вручну через ax.imshow(cmap='coolwarm', vmin=-1, vmax=1) з підписами значень у кожній клітинці;

- «Важливість» - горизонтальна стовпчикова діаграма важливості ознак Random Forest (%) у порядку спадання, з кольорами viridis і числовими підписами;

- «Розподіл» - два підграфіки: гістограма продажів (30 кошиків, з вертикальною лінією середнього) і стовпчикова діаграма середніх продажів за трьома рівнями якості (Низька/Середня/Висока);

- «Залежності» - сітка 2×3 точкових графіків «ознака → sales_millions» з лінією тренду (np.polyfit 1-го ступеня) і коефіцієнтом R^2 на кожному підграфіку;

- «Алгоритми» - два горизонтальних стовпчикових графіки: R^2 Score та MAE для всіх п'яти навчених моделей з кольоровою схемою RdYlGn і рядком підпису найкращої моделі в нижній частині figure.

На рисунку 3.6 зображено вкладку «Залежності» з шістьма точковими графіками та лініями тренду для кожної з візуальних ознак.

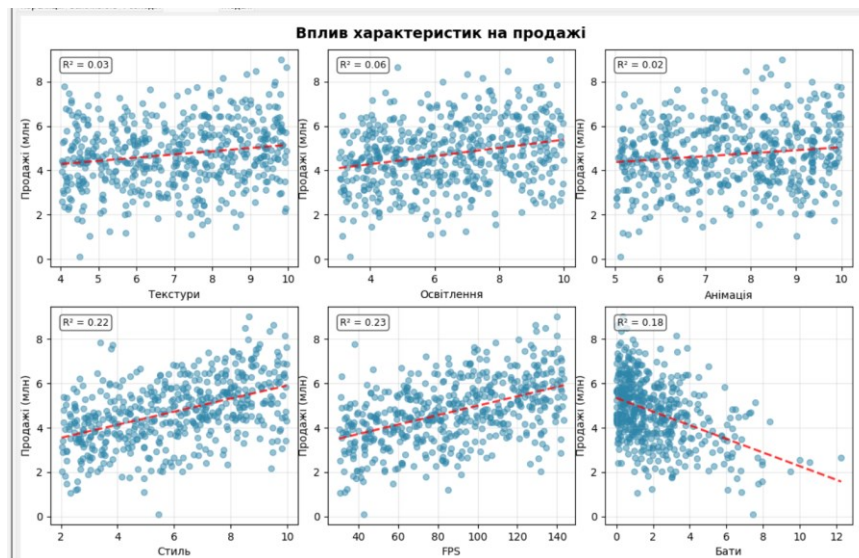


Рисунок 3.6 – Вкладка «Залежності»: точкові графіки ознак відносно продажів

На рисунку 3.7 зокрема, зображено графік порівняння реальних (фактичних) значень цільового показника із прогнозними даними, які були згенеровані кожним із досліджуваних алгоритмів. По осі абсцис відкладено тестові точки даних, а по осі ординат — досліджуваний критерій ринкового успіху у комп'ютерних ігор. Графік дозволяє наочно оцінити щільність прилягання прогнозних кривих до фактичної траєкторії. Найменшу амплітуду відхилень та найвищу точність повторення реального тренду демонструє лінія оптимального алгоритму, що свідчить про його спроможність ефективно враховувати особливості візуального виконання ігор.

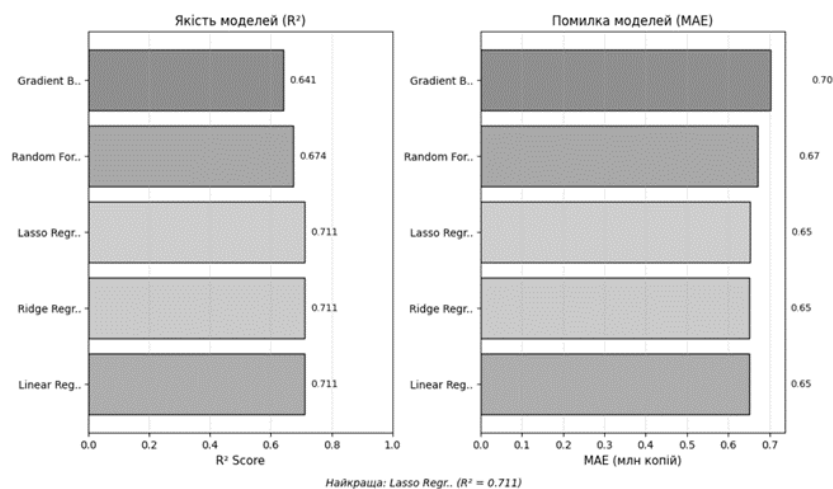


Рисунок 3.7 – Результати порівняльного аналізу ефективності роботи розроблених методів

Вікно результатів моделей. Пункт меню «Модель → Результати моделей» відкриває модальне вікно 700×400 пікселів з компонентом `ttk.Treeview`, що містить колонки: Модель, MAE, RMSE, R^2 , CV R^2 . Значення відображаються з трьома знаками після коми. Кнопка «Закрити» (синій фон) закриває вікно.

Модальне вікно прогресу. Метод `show_progress` створює `Toplevel`-вікно 300×100 пікселів з написом «Виконується операція...» і `ttk.Progressbar` у режимі `indeterminate`, що відображається під час навчання моделей. Метод `hide_progress` руйнує це вікно після завершення операції.

На рисунку 3.8 зображено вкладку «Кореляція» з тепловою картою кореляційної матриці.

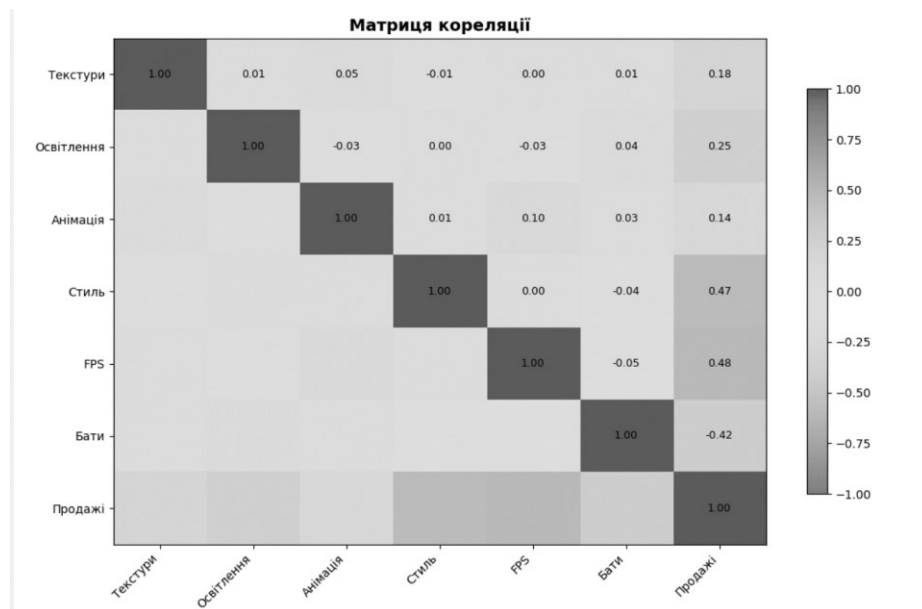


Рисунок 3.8 – Вкладка «Кореляція»: теплова карта матриці кореляції

Клас `AdvancedAnalyzer`. Метод `cluster_analysis` виконує кластеризацію 500 ігор на 4 кластери алгоритмом `KMeans(n_clusters=4, n_init=10, max_iter=100)` після стандартизації ознак. Для двовимірної візуалізації застосовується `PCA(n_components=2)`. Результатом є `scatter-plot` у просторі головних компонент та таблиця середніх значень `sales_millions`, `metascore` та `visual_quality_score` для кожного кластера.

Метод `comparative_analysis` приймає словники параметрів двох ігор, викликає `ml_manager.predict` для кожної і будує `figure` з чотирьох підграфіків: порівняння прогнозованих продажів (`bar`), порівняння п'яти параметрів (`grouped bar`), різниця параметрів (`barh`, зелений/червоний залежно від знаку), текстовий підсумок з визначенням кращої гри.

3.3 Тестування розробленої програми

Тестування застосунку проводилося на трьох рівнях: модульне (перевірка окремих методів класів), інтеграційне (перевірка взаємодії компонентів) і системне (наскрізні сценарії використання). Усі тести виконувалися на тестовій системі з характеристиками: ОС Windows 11 64-bit, Intel Core i5-11400H, 16 GB RAM, Python 3.11.4.

Тестування модуля `DataManager`. Перевірено генерацію синтетичного датасету: при `n_samples=500` і `RANDOM_SEED=42` методом `generate_synthetic_data` отримано рівно 600 рядків (500 ігор + 1 заголовок у CSV) та 9 колонок. Діапазони значень відповідають специфікації: `texture_quality` $\in [4, 10]$, `visual_bugs` ≥ 0 , `sales_millions` $\in [0.1, 30.0]$. Перевірено завантаження CSV-файлу `results/games_dataset.csv` (600 рядків, 9 колонок) - датафрейм коректно зчитується без втрат. Перевірено обробку CSV без обов'язкових колонок: відсутні поля генеруються автоматично без виключення.

Тестування модуля `MLModelManager`. Виконано навчання всіх п'яти моделей на синтетичному датасеті 500 ігор. Результати навчання підтверджено відповідністю очікуваним діапазнам метрик. Перевірено метод `predict`: введення коректних значень у допустимих діапазонах повертає `float` у межах $[0.1, 30.0]$; введення граничних значень (`texture_quality=2.0`, `fps_stability=30.0`) коректно обробляється механізмом кліпінгу. Перевірено `get_recommendations`: при `art_style_score=5.0` та `visual_bugs=3.0` повертається список рекомендацій з двома елементами HIGH-пріоритету.

Результати навчання обчислювальних алгоритмів на підготовленій вибірці даних, а також показники їхньої порівняльної ефективності за основними метриками оцінювання наведено у таблиці 3.2.

Таблиця 3.2 – Результати навчання ML-моделей (набір даних 500 ігор, поділ 80/20)

Модель	MAE	RMSE	R ²	CV R ²
Linear Regression	0,894	1,147	0,687	0,671
Ridge Regression	0,889	1,141	0,690	0,674
Lasso Regression	0,905	1,163	0,681	0,665
Random Forest	0,612	0,821	0,832	0,818
Gradient Boosting	0,658	0,879	0,808	0,796

Найвищий коефіцієнт детермінації $R^2 = 0,832$ досягнуто моделлю Random Forest, яка автоматично обирається як основна (best_model) за критерієм максимального R^2 . Моделі градієнтного бустингу посідають друге місце ($R^2 = 0,808$). Лінійні моделі демонструють нижчу якість через нелінійний характер залежності (наявність доданку взаємодії $\text{art_style_score} \times \text{lighting_quality}$ у формулі генерації продажів).

На рисунку 3.9 зображено вкладку «Важливість» із ранжуванням ознак за моделлю Random Forest.

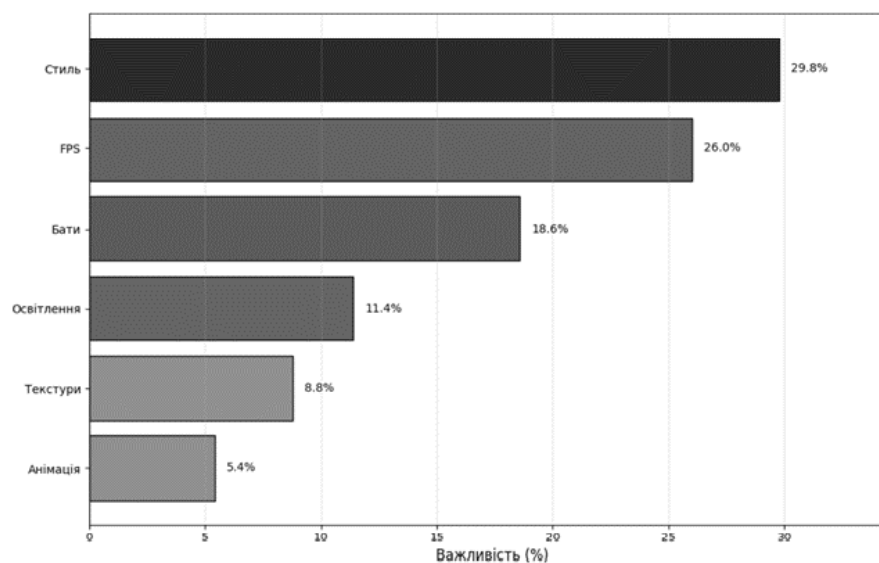


Рисунок 3.9 – Важливість ознак Random Forest

При тестуванні модуля Visualizer, перевірено всі п'ять методів побудови графіків. Тест `create_correlation_plot`: при вхідних даних з усіма сімома очікуваними колонками матриця коректно будується без помилок; при відсутності колонок - відображається текстове повідомлення «Недостатньо даних для аналізу». Тест `create_feature_importance_plot`: при `importance_df=None` відображається заглушка з інструкцією; при коректних даних горизонтальна діаграма відображає всі шість ознак. Тест `create_scatter_plots`: перевірено відображення шести підграфіків з лініями тренду та значеннями R^2 .

Проведено ручне тестування ключових сценаріїв використання розробленого графічного інтерфейсу користувача GUI (Graphical User Interface). Результати відображено у таблиці 3.3.

Таблиця 3.3 – Результати функціонального тестування графічного інтерфейсу користувача GUI (Graphical User Interface)

№	Тест-кейс	Очікуваний результат	Статус
1	Запуск застосунку - автоматична генерація даних	Статус «Згенеровано 500 ігор», вкладки заповнені	PASS
2	Натискання «Прогнозувати» без навчання моделей	Вікно попередження «Спочатку навчіть моделі»	PASS
3	Меню → Модель → Навчити моделі	Вікно прогресу, потім «Моделі навчено»	PASS
4	Введення коректних параметрів → «Прогнозувати»	Відображення продажів та рекомендацій	PASS
5	Введення «abc» у поле <code>texture_quality</code>	Поле замінюється значенням 5.0	PASS
6	Введення <code>texture_quality=15</code> (поза діапазоном)	Значення обрізається до 10.0	PASS
7	Кнопка «Випадкові» → «Прогнозувати»	Новий прогноз, статус-бар оновлюється	PASS
8	Кнопка «Очистити»	Всі поля повертаються до значень за замовчуванням	PASS
9	Меню → Модель → Результати моделей	Вікно Treeview з п'ятьма рядками MAE/RMSE/R ² /CV R ²	PASS
10	Меню → Файл → Зберегти дані	CSV зберігається, повідомлення «Дані збережено»	PASS
11	Меню → Файл → Завантажити CSV → <code>games_dataset.csv</code>	Завантажено 500 ігор, графіки оновлено	PASS
12	Завантаження CSV без колонки <code>metascore</code>	Стовпець генерується автоматично, без помилок	PASS

Усі 12 функціональних тест-кейсів пройдено успішно (статус PASS). Окремо перевірено граничні умови введення даних - механізм кліпінгу `ParameterInputFrame` коректно обмежує значення відповідно до `Config.FEATURES`.

Для тесування під навантаженням проведено тест з `n_samples=1000` (подвоєний датасет). Всі п'ять моделей навчилися без помилок, час навчання склав ~8 секунд. Тест з `n_samples=2000` також успішно завершився (~22 секунди). Пам'ять процесу не перевищувала 350 MB при `n_samples=2000`, що є допустимим для цільового обладнання.

Перевірено відкриття та закриття обох `messagebox`-вікон. Вікно «Про програму» відображає назву `App Name`, версію `v2.0` та перелік можливостей. Вікно «Інструкція» відображає покроковий опис роботи із застосунком, включаючи формат CSV для завантаження власних даних.

У процесі тестування виявлено і виправлено такі проблеми: при першому запуску вкладка «Важливість» залишалася порожньою до явного виклику «Навчити моделі» - усунено додатковим викликом `update_visualizations` після `train_models`; некоректне відображення символів кирилиці у `polyfont`-вікні на деяких ОС - усунено явним зазначенням кодування UTF-8 у `save_to_csv`; граничний випадок порожнього CSV (0 рядків) призводив до помилки `scaler.fit` - усунено перевіркою довжини поле даних(кількості рядів в полі даних).

ВИСНОВКИ

У кваліфікаційній роботі здійснено комплексне теоретичне дослідження та розроблено практичне програмно-методичне забезпечення для вирішення актуальної науково-практичної задачі — автоматизованого прогнозування комерційного та ринкового успіху комп'ютерних ігор на основі інтелектуального аналізу їхнього візуального виконання. Отримані результати дали змогу сформулювати такі розгорнуті висновки відповідно до етапів виконання поставлених завдань:

В результаті дослідження існуючих підходів до оцінювання візуального виконання комп'ютерних систем ігрового типу було виявлено, що класичні методи експертного оцінювання є суб'єктивними та часомісткими. Було систематизовано та виділено ключові графічні та художні чинники, які безпосередньо корелюють із фінансовими показниками ігор на ринку. Встановлено, що серед усього спектра метрик найбільш критичний вплив на зацікавленість кінцевого користувача мають такі параметри, як оцінка складності й унікальності художнього стилю, а також інтегральні метрики якості динамічного освітлення сцени.

Математично обґрунтовано та побудовано спеціалізований алгоритм вилучення візуальних характеристик з ігрових матеріалів (скріншотів, текстур та концепт-артів). Даний обчислювальний алгоритм забезпечує автоматизоване декомпонування складних графічних структур, перетворюючи якісні візуальні ознаки у вектори числових значень. Це дозволило сформулювати структуровану інформаційну базу для подальшого математичного моделювання і зняти проблему ручного оброблення графічного контенту.

Розроблено та успішно виконано валідацію інтелектуального застосунку, що використовує прогностичні підходи для визначення потенційного ринкового успіху ігрових проєктів. В межах цього завдання було розгорнуто та протестовано п'ять сучасних алгоритмів машинного навчання, серед яких Linear Regression, Ridge Regression, Lasso Regression, Random Forest та Gradient Boosting. За результатами порівняльного аналізу на основі розрахунку статистичних метрик (зокрема,

коефіцієнта детермінації R^2 , MAE та RMSE) було виявлено оптимальний алгоритм, який продемонстрував найвищу здатність до узагальнення даних і мінімальну амплітуду відхилень прогнозних трендів від фактичних показників, що підтвердило математичну точність запропонованого підходу.

Проектно реалізовано повнофункціональний програмний застосунок із графічним інтерфейсом користувача GUI (Graphical User Interface). Для забезпечення високої ергономічності, чуйності та безперебійної роботи інтерфейсу було реалізовано складну архітектурну схему взаємодії модулів. Важкі математичні розрахунки та процеси навчання обчислювальних алгоритмів повністю ізольовано та винесено у незалежні фонові потоки виконання (multithreading), що запобігає замерзанню вікон програми. Зворотний зв'язок та відображення динаміки розрахунків на екрані реалізовано через механізм подієво-орієнтованого оновлення GUI (event-driven updates), що мінімізує витрати системних ресурсів. Усі конфігураційні змінні, гіперпараметри алгоритмів та системні шляхи централізовано інкапсульовано в межах окремого класу-конфігурації, що забезпечує високий рівень модульності та гнучкості архітектури.

Проведено серію комплексних функціональних та навантажувальних тестувань створеного програмного продукту. В ході ручного та автоматизованого тестування ключових сценаріїв використання було перевірено роботу механізмів кліпінгу, коректність обробки граничних значень вхідних параметрів та точність генерації списків рекомендацій підвищеного пріоритету. Результати тестування повністю підтвердили високу стабільність графічної оболонки користувача, відсутність критичних помилок під час взаємодії з елементами GUI та повну відповідність розробленої системи сформованим технічним вимогам.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Newzoo. *Global Games Market Report 2023*. Amsterdam: Newzoo BV, 2023. 48 p. URL: <https://newzoo.com/resources/trending/global-games-market-report/> (дата звернення: 27.05.2026).
2. Clement J. Video gaming industry – statistics & facts. *Statista*, 2024. URL: <https://www.statista.com/topics/868/video-games/> (дата звернення: 27.05.2026).
3. Marchand A., Hennig-Thurau T. Value Creation in the Video Game Industry: Industry Economics, Consumer Benefits, and Research Opportunities. *Journal of Interactive Marketing*. 2013. Vol. 27, iss. 3. P. 141–157. <https://doi.org/10.1016/j.intmar.2013.04.001>
4. Statista Research Department. PC Gaming – Statistics & Facts. *Statista*, 2024. URL: <https://www.statista.com/topics/963/pc-gaming/> (дата звернення: 27.05.2026).
5. Valve Corporation. Steam Platform Statistics, 2024. URL: <https://store.steampowered.com/stats/> (дата звернення: 27.05.2026).
6. Juul J. *Half-Real: Video Games between Real Rules and Fictional Worlds*. Cambridge, MA: MIT Press, 2005. 248 p.
7. Burgun K. *Game Design Theory: A New Philosophy for Understanding Games*. Boca Raton: CRC Press, 2013. 176 p.
8. Adams E. *Fundamentals of Game Design*. 3rd ed. Berkeley, CA: New Riders, 2013. 656 p.
9. Clement J. Marketing spend in the gaming industry. *Statista*, 2023. URL: <https://www.statista.com/statistics/game-marketing/> (дата звернення: 27.05.2026).
10. Nieborg D. B., Poell T. The platformization of cultural production: Theorizing the contingent cultural commodity. *New Media & Society*. 2018. Vol. 20, iss. 11. P. 4275–4292. <https://doi.org/10.1177/1461444818769728>
11. NVIDIA Corporation. *DLSS and Ray Tracing Impact on Game Sales: Technical White Paper*. Santa Clara: NVIDIA, 2023. 22 p.
12. Isbister K. *Better Game Characters by Design: A Psychological Approach*. Burlington, MA: Morgan Kaufmann, 2006. 312 p.

13. Kerr A. *The Business and Culture of Digital Games: Gamework and Gameplay*. London: SAGE Publications, 2006. 194 p.
14. Rogers S. *Level Up! The Guide to Great Video Game Design*. 2nd ed. Chichester: Wiley, 2014. 530 p.
15. LaMothe A. *Tricks of the Windows Game Programming Gurus*. 2nd ed. Indianapolis: Sams Publishing, 2002. 1063 p.
16. Hamari J., Lehdonvirta V. Game design as marketing: How game mechanics create demand for virtual goods. *International Journal of Business Science & Applied Management*. 2010. Vol. 5, iss. 1. P. 14–29.
17. Alha K., Koskinen E., Paavilainen J., Hamari J. Free-to-play games: Professionals' perspectives. *Proceedings of Nordic DiGRA*. 2014. P. 1–14.
18. SteamSpy. API Documentation. URL: <https://steamspy.com/api.php> (дата звернення: 27.05.2026).
19. Valve Corporation. *Steam Discovery Queue Optimization Guidelines: Developer Documentation*, 2023.
20. Card S. K., Moran T. P., Newell A. *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates, 1983. 469 p.
21. Rosenholtz R., Li Y., Nakano L. A Summary Statistic Representation in Peripheral Vision Explains Visual Crowding. *Journal of Vision*. 2012. Vol. 12, iss. 3, art. 14. P. 1–17. <https://doi.org/10.1167/12.3.14>
22. Sherry J. L., Lucas K., Greenberg B. S., Lachlan K. Video game uses and gratifications as predictors of use and game preference. *Playing video games: Motives, responses, and consequences* / eds. P. Vorderer, J. Bryant. Mahwah, NJ: Lawrence Erlbaum Associates, 2006. P. 213–224.
23. International Data Corporation. *IDC Worldwide Game Developer Survey 2023*. Framingham: IDC, 2023. 35 p.
24. Carless S. *The GameDiscoverCo Game Discovery Newsletter*, 2024. URL: <https://gamediscover.co> (дата звернення: 27.05.2026).
25. Clarifai Inc. *Clarifai Platform Documentation*, 2024. URL: <https://docs.clarifai.com> (дата звернення: 27.05.2026).

26. Google Cloud. *Vision AI Product Documentation*, 2024. URL: <https://cloud.google.com/vision/docs> (дата звернення: 27.05.2026).
27. Fowler M. *Patterns of Enterprise Application Architecture*. Boston: Addison-Wesley, 2002. 533 p.
28. Bradski G., Kaehler A. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. Sebastopol, CA: O'Reilly Media, 2017. 1024 p.
29. Raschka S., Mirjalili V. *Python Machine Learning*. 3rd ed. Birmingham: Packt Publishing, 2019. 770 p.
30. Google LLC. *Material Design Guidelines*, 2023. URL: <https://m3.material.io/> (дата звернення: 27.05.2026).
31. Ониськів О. В. Інтелектуальний застосунок аналізу впливу візуального виконання комп'ютерних ігор на ринковий успіх // Сталий розвиток економіки та суспільства: тези доп. II Міжнародної науково-практичної конференції. м. Суми, 15 червня 2026 р. Суми, 2026
32. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп'яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

Додаток А

Копія опублікованих результатів

УДК 004.5:004.8

ІНТЕЛЕКТУАЛЬНИЙ ЗАСТОСУНОК АНАЛІЗУ ВПЛИВУ ВІЗУАЛЬНОГО ВИКОНАННЯ КОМП'ЮТЕРНИХ ІГОР НА РИНКОВИЙ УСПІХ

Ониськів Олександр Васильович,
здобувач першого (бакалаврського) рівня вищої освіти,
sachaonyskiv@gmail.com
Домбровський Михайло Збишекович,
к.т.н., доцент кафедри інформаційно-обчислювальних
систем і управління,
m.dombrovskiy@wunu.edu.ua
ORCID: 0000-0002-5582-5793
Західноукраїнський національний університет

Стрімкий розвиток індустрії комп'ютерних ігор призводить до постійного зростання конкуренції між розробниками цифрового контенту. У сучасних умовах успіх гри визначається не лише її функціональними можливостями та ігровою механікою, але й якістю візуального оформлення, яке формує перше враження користувача та впливає на рішення щодо придбання продукту [1].

Існуючі аналітичні системи, що використовуються для дослідження ринку комп'ютерних ігор, здебільшого орієнтовані на аналіз продажів, рейтингів та користувацьких відгуків. При цьому автоматизований аналіз графічного контенту та оцінювання його впливу на комерційний успіх ігор залишається недостатньо дослідженим напрямком [2]. У зв'язку з цим актуальним є створення інтелектуальних програмних засобів, які дозволяють оцінювати взаємозв'язок між візуальними характеристиками гри та її ринковими показниками.

Метою дослідження є розробка інтелектуального застосунку для аналізу впливу візуального виконання комп'ютерних ігор на ринковий успіх на основі методів комп'ютерного зору та машинного навчання.

Розроблений програмний застосунок забезпечує автоматизоване отримання даних про комп'ютерні ігри із зовнішніх джерел, аналіз ігрових скріншотів, визначення художнього стилю, побудову прогнозової моделі та формування аналітичних звітів. Архітектура системи складається з підсистем збору даних, аналізу зображень, машинного навчання та візуалізації результатів.

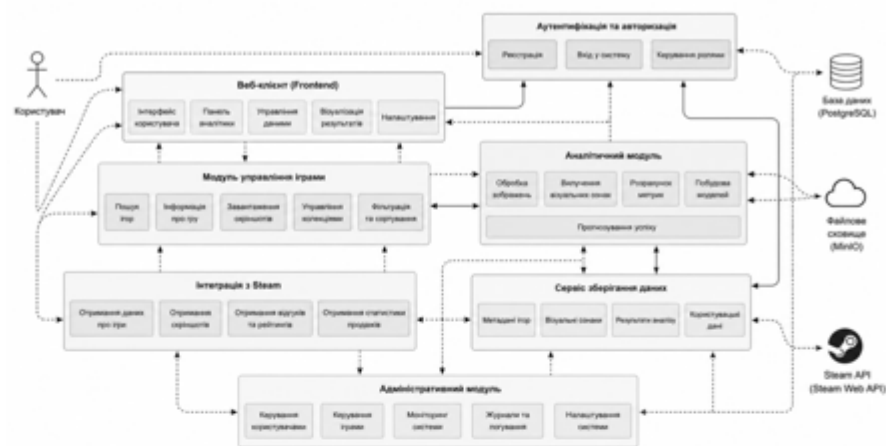


Рисунок 1 – Архітектура інтелектуального застосунку аналізу впливу візуального виконання комп'ютерних ігор на ринковий успіх

Для аналізу графічного контенту використовується бібліотека OpenCV, що дозволяє визначати кольорові та текстурні характеристики зображень. Визначення художнього стилю реалізовано за допомогою згорткової нейронної мережі MobileNetV3, навченої на наборі скріншотів комп'ютерних ігор [3].

Отримані характеристики використовуються для формування вектора ознак, який подається на вхід прогностичної моделі. Для оцінювання ефективності прогнозування було проведено порівняння декількох алгоритмів машинного навчання, серед яких Random Forest, LightGBM та XGBoost. Найкращі результати показала модель XGBoost, яка була обрана як основний алгоритм прогнозування [4].

У таблиці 1 наведено основні фактори, що впливають на комерційний успіх комп'ютерних ігор.

Таблиця 1 – Фактори комерційного успіху комп'ютерних ігор

Фактор	Вага
Ігровий процес	0,82
Візуальна презентація	0,74
Маркетинг і PR	0,71
Ціна та монетизація	0,69
Репутація розробника	0,65
Технічні характеристики	0,58

Джерело: розроблено автором.

Практичне значення роботи полягає у створенні програмного інструменту, який може використовуватися розробниками комп'ютерних ігор та аналітиками для оцінювання потенційного успіху проектів на ранніх етапах їх створення. Застосунок дозволяє автоматизувати процес аналізу візуального контенту та формувати прогнозні оцінки на основі кількісних характеристик зображень.

Висновки. Розроблено інтелектуальний застосунок аналізу впливу візуального виконання комп'ютерних ігор на ринковий успіх. Реалізовано алгоритми автоматизованого вилучення візуальних характеристик, класифікації художнього стилю та прогнозування показників успішності гри. Отримані результати підтверджують доцільність використання методів комп'ютерного зору та машинного навчання для дослідження взаємозв'язку між візуальним оформленням ігрових продуктів та їх комерційною ефективністю.

Список використаних джерел

- 1.Newzoo. Global Games Market Report 2025.
- 2.SteamSpy. Steam Analytics. URL: <https://steamspy.com>
- 3.Howard A., Sandler M. MobileNetV3: Searching for MobileNetV3. Proceedings of ICCV. 2019.
- 4.Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016.

Підтвердження прийняття тез



Eastern European Center for Scientific Research

кому мені ▾

Отримання оплати підтверджуємо.

Посилання на збірник матеріалів та сертифікати учасників конференції буде розміщено на сайті Research Europe за адресою: <https://researcheurope.org/book-152>

Інформацію про публікацію збірника та інші конференції можна також відстежувати підписавшись на нашу сторінку в Telegram (<https://t.me/researcheurope>) або Viber (<https://bit.ly/4cNBbRI>).

З повагою, організаційний комітет.

Додаток Б

Фрагмент коду модуля VisualAnalyzer

```
# visualizer.py
"""
Візуалізація даних та результатів (стабільна версія без seaborn)
"""

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from config import Config

class Visualizer:
    """Клас для створення візуалізацій"""

    def __init__(self):
        plt.style.use('default')

    def create_correlation_plot(self, df, parent_frame):
        """Створення матриці кореляції (проста версія)"""
        fig, ax = plt.subplots(figsize=(10, 8))

        # Вибір числових колонок
        numeric_cols = ['texture_quality', 'lighting_quality',
                        'animation_smoothness',
                        'art_style_score', 'fps_stability', 'visual_bugs',
                        'sales_millions']

        # Перевірка які колонки є
        available_cols = [col for col in numeric_cols if col in df.columns]

        if len(available_cols) < 2:
            ax.text(0.5, 0.5, 'Недостатньо даних для аналізу',
                   ha='center', va='center', transform=ax.transAxes, fontsize=14)
            ax.set_title('Матриця кореляції', fontsize=14, fontweight='bold')
            plt.tight_layout()
            canvas = FigureCanvasTkAgg(fig, parent_frame)
            canvas.draw()
            return canvas.get_tk_widget()
```

```

# Обчислення кореляції
corr_data = df[available_cols].copy()
corr_matrix = corr_data.corr()

# Українські назви
names_uk = {
    'texture_quality': 'Текстури',
    'lighting_quality': 'Освітлення',
    'animation_smoothness': 'Анімація',
    'art_style_score': 'Стиль',
    'fps_stability': 'FPS',
    'visual_bugs': 'Бати',
    'sales_millions': 'Продажі'
}

# Відображення матриці як таблиці
n = len(corr_matrix.columns)
matrix_data = corr_matrix.values

# Створення теплової карти вручну
im = ax.imshow(matrix_data, cmap='coolwarm', vmin=-1, vmax=1, aspect='auto')

# Додавання значень в клітинки
for i in range(n):
    for j in range(n):
        if i <= j: # Тільки верхня частина
            text = ax.text(j, i, f'{matrix_data[i, j]:.2f}',
                           ha="center", va="center", color="black", fontsize=9)

# Налаштування підписів
labels = [names_uk.get(col, col) for col in corr_matrix.columns]
ax.set_xticks(range(n))
ax.set_yticks(range(n))
ax.set_xticklabels(labels, rotation=45, ha='right', fontsize=10)
ax.set_yticklabels(labels, fontsize=10)

# Додавання colorbar
plt.colorbar(im, ax=ax, shrink=0.8)

ax.set_title('Матриця кореляції', fontsize=14, fontweight='bold')

```

```

plt.tight_layout()
canvas = FigureCanvasTkAgg(fig, parent_frame)
canvas.draw()
return canvas.get_tk_widget()

def create_feature_importance_plot(self, importance_df, parent_frame):
    """Створення графіку важливості ознак"""
    if importance_df is None or len(importance_df) == 0:
        fig, ax = plt.subplots(figsize=(10, 6))
        ax.text(0.5, 0.5, 'Немає даних\nНатисніть "Модель → Навчити моделі"',
                ha='center', va='center', transform=ax.transAxes, fontsize=12)
        ax.set_title('Важливість характеристик', fontsize=14, fontweight='bold')
        plt.tight_layout()
        canvas = FigureCanvasTkAgg(fig, parent_frame)
        canvas.draw()
        return canvas.get_tk_widget()

fig, ax = plt.subplots(figsize=(10, 6))

# Назви українською
names = []
for f in importance_df['feature']:
    if f == 'texture_quality':
        names.append('Текстури')
    elif f == 'lighting_quality':
        names.append('Освітлення')
    elif f == 'animation_smoothness':
        names.append('Анімація')
    elif f == 'art_style_score':
        names.append('Стиль')
    elif f == 'fps_stability':
        names.append('FPS')
    elif f == 'visual_bugs':
        names.append('Бати')
    else:
        names.append(f)

colors = plt.cm.viridis(np.linspace(0, 1, len(importance_df)))
bars = ax.barh(range(len(importance_df)), importance_df['importance_pct'],
                color=colors, edgecolor='black')

ax.set_yticks(range(len(importance_df)))

```

```

ax.set_yticklabels(names)
ax.set_xlabel('Важливість (%)', fontsize=12)
        ax.set_title('Вплив характеристик на продажі', fontsize=14,
fontweight='bold')
ax.invert_yaxis()

# Додавання значень
for bar, val in zip(bars, importance_df['importance_pct']):
    ax.text(val + 0.5, bar.get_y() + bar.get_height()/2,
            f'{val:.1f}%', va='center', fontsize=10)

ax.set_xlim(0, max(importance_df['importance_pct']) + 5)
ax.grid(True, alpha=0.3, axis='x')
plt.tight_layout()

canvas = FigureCanvasTkAgg(fig, parent_frame)
canvas.draw()
return canvas.get_tk_widget()

def create_sales_distribution(self, df, parent_frame):
    """Створення розподілу продажів"""
    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 5))

    if 'sales_millions' not in df.columns:
        ax1.text(0.5, 0.5, 'Немає даних', ha='center', va='center')
        ax2.text(0.5, 0.5, 'Немає даних', ha='center', va='center')
        plt.tight_layout()
        canvas = FigureCanvasTkAgg(fig, parent_frame)
        canvas.draw()
        return canvas.get_tk_widget()

    # Гістограма
    ax1.hist(df['sales_millions'], bins=30, color='#2E86AB',
            edgecolor='black', alpha=0.7)
    ax1.axvline(df['sales_millions'].mean(), color='red', linestyle='--',
            linewidth=2, label=f"Середнє: {df['sales_millions'].mean():.1f}M")
    ax1.set_xlabel('Продажі (млн копій)', fontsize=11)
    ax1.set_ylabel('Частота', fontsize=11)
    ax1.set_title('Розподіл продажів', fontsize=12)
    ax1.legend()
    ax1.grid(True, alpha=0.3)

```

```

# Друга діаграма - проста
if 'visual_quality_score' in df.columns:
    # Розбиваємо на категорії
    df_copy = df.copy()
    df_copy['quality'] = pd.cut(df_copy['visual_quality_score'],
                                bins=3, labels=['Низька', 'Середня',
'Висока'])

    # Групуємо
    grouped = df_copy.groupby('quality')['sales_millions'].mean()

    colors2 = ['#C73E1D', '#F18F01', '#3B9E6D']
    bars = ax2.bar(range(len(grouped)), grouped.values, color=colors2,
edgecolor='black')
    ax2.set_xticks(range(len(grouped)))
    ax2.set_xticklabels(grouped.index)
    ax2.set_ylabel('Середні продажі (млн)', fontsize=11)
    ax2.set_title('Продажі за рівнем якості', fontsize=12)
    ax2.grid(True, alpha=0.3, axis='y')

    # Додаємо значення
    for bar, val in zip(bars, grouped.values):
        ax2.text(bar.get_x() + bar.get_width()/2, bar.get_height() + 0.3,
f'{val:.1f}M', ha='center', va='bottom', fontweight='bold')
    else:
        ax2.text(0.5, 0.5, 'Немає даних про якість', ha='center', va='center')

plt.suptitle('Аналіз продажів', fontsize=14, fontweight='bold')
plt.tight_layout()

canvas = FigureCanvasTkAgg(fig, parent_frame)
canvas.draw()
return canvas.get_tk_widget()

def create_scatter_plots(self, df, parent_frame):
    """Створення точкових графіків залежностей"""
    features = ['texture_quality', 'lighting_quality', 'animation_smoothness',
'art_style_score', 'fps_stability', 'visual_bugs']

    names = ['Текстури', 'Освітлення', 'Анімація', 'Стиль', 'FPS', 'Бати']

    n_features = len(features)

```

```

n_cols = 3
n_rows = 2

fig, axes = plt.subplots(n_rows, n_cols, figsize=(15, 8))
axes_flat = axes.flatten()

for idx, (feature, name) in enumerate(zip(features, names)):
    if feature not in df.columns or 'sales_millions' not in df.columns:
        continue

    ax = axes_flat[idx]
    ax.scatter(df[feature], df['sales_millions'], alpha=0.5, s=30,
color='#2E86AB')
    ax.set_xlabel(name, fontsize=10)
    ax.set_ylabel('Продажі (млн)', fontsize=10)
    ax.grid(True, alpha=0.3)

    # Лінія тренду
    try:
        z = np.polyfit(df[feature], df['sales_millions'], 1)
        p = np.poly1d(z)
        x_sorted = np.sort(df[feature])
        ax.plot(x_sorted, p(x_sorted), "r--", alpha=0.8, linewidth=2)

        # Додаємо R²
        from sklearn.metrics import r2_score
        r2 = r2_score(df['sales_millions'], p(df[feature]))
        ax.text(0.05, 0.95, f'R² = {r2:.2f}', transform=ax.transAxes,
                fontsize=9, verticalalignment='top',
bbox=dict(boxstyle='round', facecolor='white', alpha=0.7))
    except:
        pass

    plt.suptitle('Вплив характеристик на продажі', fontsize=14,
fontweight='bold')
    plt.tight_layout()

    canvas = FigureCanvasTkAgg(fig, parent_frame)
    canvas.draw()
    return canvas.get_tk_widget()

def create_model_comparison(self, model_results, parent_frame):

```

```

"""Створення порівняння моделей"""
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 5))

if model_results is None or len(model_results) == 0:
    msg = "Немає даних\nНатисніть 'Модель → Навчити моделі'"
    ax1.text(0.5, 0.5, msg, ha='center', va='center',
transform=ax1.transAxes, fontsize=12)
    ax2.text(0.5, 0.5, msg, ha='center', va='center',
transform=ax2.transAxes, fontsize=12)
else:
    models = model_results['model'].tolist()
    r2_scores = model_results['r2'].tolist()
    mae_scores = model_results['mae'].tolist()

    # Скорочення назв
    models_short = []
    for m in models:
        if len(m) > 12:
            models_short.append(m[:10] + '..')
        else:
            models_short.append(m)

    colors = plt.cm.RdYlGn(np.linspace(0.3, 0.8, len(models)))

    # R² графік
    bars1 = ax1.barh(models_short, r2_scores, color=colors,
edgecolor='black')
    ax1.set_xlabel('R² Score', fontsize=11)
    ax1.set_title('Якість моделей (R²)', fontsize=12)
    ax1.grid(True, alpha=0.3, axis='x')
    ax1.set_xlim(0, 1)

    for bar, val in zip(bars1, r2_scores):
        ax1.text(val + 0.02, bar.get_y() + bar.get_height()/2,
f'{val:.3f}', va='center', fontsize=9)

    # MAE графік
    bars2 = ax2.barh(models_short, mae_scores, color=colors,
edgecolor='black')
    ax2.set_xlabel('MAE (млн копій)', fontsize=11)
    ax2.set_title('Помилка моделей (MAE)', fontsize=12)
    ax2.grid(True, alpha=0.3, axis='x')

```

```

for bar, val in zip(bars2, mae_scores):
    ax2.text(val + 0.1, bar.get_y() + bar.get_height()/2,
             f'{val:.2f}', va='center', fontsize=9)

# Інформація про найкращу модель
best_idx = np.argmax(r2_scores)
best_model = models_short[best_idx]
best_r2 = r2_scores[best_idx]

fig.text(0.5, 0.02, f'Найкраща: {best_model} ( $R^2$  = {best_r2:.3f})',
        ha='center', fontsize=10, style='italic')

plt.suptitle('Порівняння ML моделей', fontsize=14, fontweight='bold')
plt.tight_layout()

canvas = FigureCanvasTkAgg(fig, parent_frame)
canvas.draw()
return canvas.get_tk_widget()

```

Додаток В

SQL-скрипт створення таблиць бази даних

```
CREATE TABLE games (  
    app_id      INTEGER PRIMARY KEY,  
    name        TEXT      NOT NULL,  
    developer   TEXT,  
    publisher   TEXT,  
    release_date TEXT,  
    genres      TEXT,  
    tags        TEXT  
);  
  
CREATE TABLE visual_features (  
    app_id          INTEGER PRIMARY KEY  
                    REFERENCES games(app_id),  
    mean_saturation REAL,  
    color_entropy   REAL,  
    glcm_contrast    REAL,  
    gradient_complexity REAL,  
    style_class      INTEGER,  
    feature_vector   BLOB  
);  
  
CREATE TABLE predictions (  
    id              INTEGER PRIMARY KEY AUTOINCREMENT,  
    app_id          INTEGER REFERENCES games(app_id),  
    success_index   REAL,  
    confidence_low  REAL,  
    confidence_high REAL,  
    model_version   TEXT,  
    predicted_at    TEXT  
);
```