МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

ХЛОПЧИК ВЛАДИСЛАВ МАКСИМОВИЧ

Допускається до захисту:
в.о. завідувача кафедри
інформаційних
технологій канд. техн.
наук, доцент _____
О. В. Зелінська
«____»_____ 20__ р.

ДОСЛІДЖЕННЯ МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ ДЛЯ ОЦІНКИ ПРОДАЖІВ ВІДЕОІГОР

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (магістерська) робота

Науковий керівник:
Т. В. Січко, доцент кафедри
інформаційних технологій
к. т. н., доцент

_____

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____

Вінниця — 2024

АНОТАЦІЯ

Хлопчик В. М. Дослідження методів інтелектуального аналізу даних для оцінки продажу відеоігор. Спеціальність 122 «Комп’ютерні науки», Освітня програма «Комп’ютерні технології обробки даних (Data science)». Донецький національний університет імені Василя Стуса, Вінниця, 2024.

У кваліфікаційній (магістерській) роботі розроблено програму для аналізу та візуалізації даних для продажу відеоігор. Показано: аналіз продажу відеоігор різними способами та за допомогою рекурентної нейронної мережі LSTM, процес розробки програми.

Ключові слова: Python, аналіз даних, оцінка продажу, методи дослідження.

Khlopchyk V. M. Research of methods of intellectual data analysis for estimation of video games sales. Specialty 122 "Computer Science", Educational program "Computer Technologies of Data Processing (Data science)". Vasyl' Stus Donetsk National University, Vinnytsia, 2023.

In the qualification (master's) work, a program for analyzing and visualizing data for video game sales was developed. Shown: analysis of video game sales in different ways and with the help of the recurrent neural network LSTM, the process of program development.

Keywords: Python, data analysis, sales estimation, research methods

ЗМІСТ

ВСТУП 4
РОЗДІЛ 1 6
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ 6
1.1 Огляд предметної галузі 6
1.2 Огляд існуючих сервісів у аналізі продажів відеоігор 7
1.3 Постановка задачі 12
РОЗДІЛ 2 14
ВИКОРИСТАНІ ТЕХНОЛОГІЇ 14
2.1 Особливості Python 14
2.2 Бібліотеки обробки даних 24
РОЗДІЛ 3 44
АНАЛІЗ ДАНИХ ДЛЯ ОЦІНКИ ПРОДАЖУ ВІДЕОІГОР 44
3.1 Аналіз датасета на мовою програмування Python 44
3.2 Інтелектуальний аналіз продажів відеоігор 57
3.3 Використані методи та аналіз датасета 63
ВИСНОВКИ 68
ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 69

ВСТУП

Актуальність теми дослідження методів аналізу даних для оцінки продажів відеоігор визначається низкою факторів. Сучасна індустрія відеоігор є однією з найбільш прогресуючих та прибуткових галузей розваг, привертаючи мільйони гравців та інвесторів по всьому світу. З цієї причини зростає потреба в глибокому аналізі даних для прийняття обґрунтованих рішень щодо ефективності маркетингових стратегій, прогнозування та адаптації до змін у попиті споживачів, що відповідають динаміці швидкозмінюючого ринку відеоігор. Оцінка продажів відеоігор має важливе значення для ігрових компаній, видавців, розробників та інвесторів, оскільки вона визначає успішність гри на ринку. Сучасні інструменти аналізу даних дозволяють виявляти ключові тенденції в галузі, визначати найбільш вигідні ринки, а також прогнозувати попит на конкретні жанри та платформи.

У сучасному світі відеоігри стали важливим елементом розваг та взаємодії. Вони не тільки задовольняють потреби гравців, але й мають глибокий культурний та економічний вплив. Розкриття потенціалу цієї індустрії стає дуже важливим, особливо з огляду на її стрімкий розвиток і величезний ринок. Використання методів інтелектуального аналізу даних і візуалізації обумовлено здатністю виявляти невидимі закономірності у великих обсягах інформації. Це дозволяє швидко і розумно приймати рішення, а також ефективно реагувати на зміни в галузі.

Метою роботи є огляд методів інтелектуального аналізу даних для прогнозування та оцінки продажів відеоігор.

Об’єкт дослідження – процес опрацювання методів аналізу продажів відеоігор, процес застосування інтелектуального аналізу.

Предмет дослідження – методи аналізу статистичних даних, які можуть використовуватись для аналізу даних для оцінки продажів відеоігор.

Завдання дослідження полягають у наступному:

- аналіз наявних даних: збір та аналіз наявних даних про продажі відеоігор, включаючи обсяги продажів, географічний розподіл та категорії ігор;
- вибір методів аналізу даних: вивчення та вибір різних методів аналізу даних, таких як статистичний аналіз, машинне навчання, глибоке навчання, кластеризація, правила асоціацій, рекомендаційні системи тощо;
- розробка моделі: розробка моделі для прогнозування та оцінки продажів відеоігор на основі обраних методів інтелектуального аналізу даних;
- висновки та рекомендації: зробити висновки щодо різних методів аналізу даних для оцінки продажів відеоігор.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд предметної галузі

Тема дослідження методів аналізу даних для оцінки продажів відеоігор є актуальною і важливою в сучасній індустрії відеоігор. Розуміння того, як аналізувати дані та використовувати їх для оцінки та прогнозування продажів, може допомогти розробникам та видавцям ігрових компаній приймати ефективні рішення, залучати аудиторію та досягати комерційного успіху.

Одним із ключових факторів успіху відеогри є її комерційна привабливість. Розробники та видавці повинні вміти прогнозувати та оцінювати потенційні продажі гри, щоб приймати рішення щодо бюджету розробки, маркетингових заходів та цінових стратегій. Аналіз даних може допомогти визначити ключові фактори, що впливають на продажі відеоігор, такі як жанр, графіка, геймплей і маркетингові кампанії.

Крім того, величезна кількість даних про продажі відеоігор, накопичених протягом багатьох років, надає можливості для дослідження та застосування різних методів аналізу даних. Статистичне моделювання, машинне навчання та інші аналітичні інструменти можуть виявити складні залежності, тенденції та кореляції, допомагаючи зрозуміти, які фактори сприяють успіху відеоігор, а які призводять до їхнього комерційного провалу.

Зростаючий вплив віртуальної реальності (VR), доповненої реальності (AR) та інших нових технологій також змінює індустрію відеоігор, вимагаючи нових підходів до аналізу даних для вимірювання продажів цих ігор. Вивчення методів аналізу даних у контексті цих ігор може допомогти виявити унікальні фактори, які впливають на успіх ігор і сприяють розвитку цих нових ринків [1].

Тому, зважаючи на постійний розвиток індустрії відеоігор та величезну кількість доступних даних, дослідження методів інтелектуального аналізу даних для оцінки продажів відеоігор залишається актуальним і має великий

потенціал для вдосконалення стратегій розробки, маркетингу та прийняття рішень у цій галузі.

1.2 Огляд існуючих сервісів аналізу продажів відеоігор

За аналог було взято веб-сервіс SteamSpy, який використовує публічні дані з профілів користувачів на платформі Steam для аналізу статистики продажів відеоігор. Основною ідеєю SteamSpy було приблизне визначення кількості проданих копій на підставі публічно доступних даних про кількість власників гри на Steam.

Сервіс SteamSpy проаналізував публічні профілі користувачів і використав відому інформацію про власників ігор, щоб оцінити загальну кількість проданих копій. Він також враховувала такі показники, як час, проведений за грою, та іншу статистику, щоб надати додаткову інформацію про популярність гри [1]. Приклад інтерфейсу зображено на рис. 1.1.

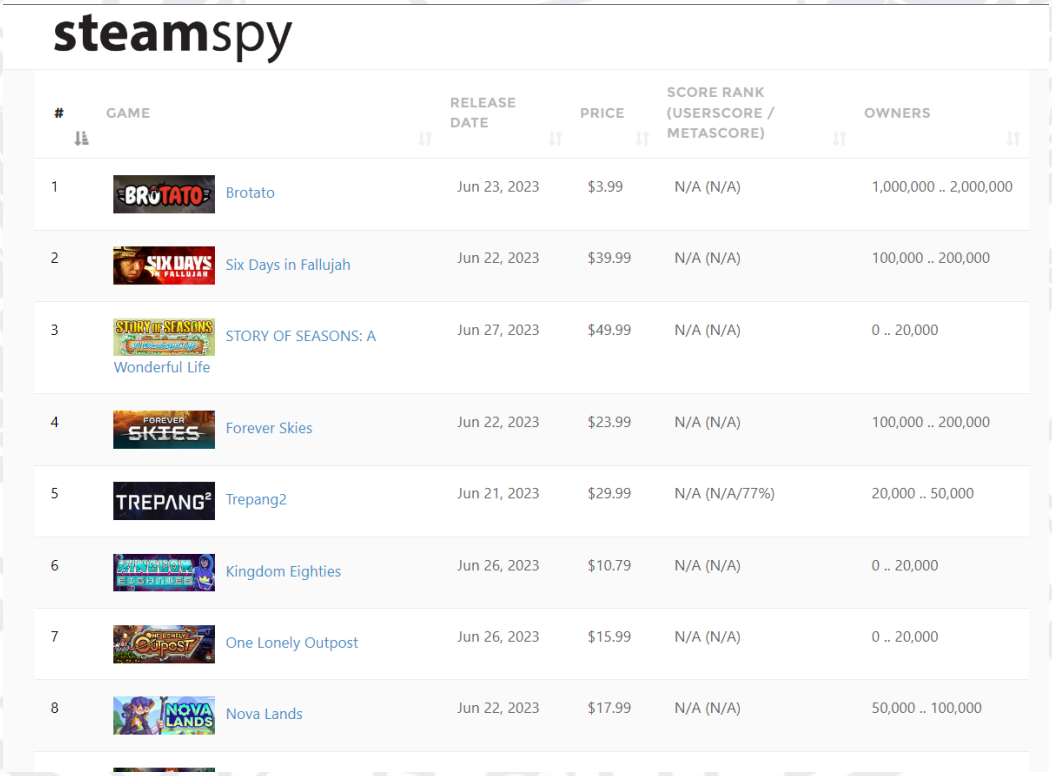

Рисунок 1.1 – Інтерфейс SteamSpy

Переваги використання SteamSpy для аналізу даних про продаж відеоігор:

- SteamSpy надає загальну оцінку продажів гри, яка є корисною для отримання загального уявлення про популярність та комерційний успіх гри;
- SteamSpy також надає іншу статистику, корисну для аналізу ринку, наприклад, середній час гри та популярність за країнами та платформами;
- SteamSpy був безкоштовним сервісом і тому був доступний широкому колу користувачів і дослідників.

До неділоків використання SteamSpy для аналізу даних про продаж відеоігор можна віднести:

- дані, отримані з SteamSpy, є наближеними, заснованими на загальнодоступних профілях користувачів, що може призвести до неточності або неповноти інформації;
- SteamSpy припинив свою роботу у 2018 році і тому не надає актуальних даних про продажі відеоігор у Steam;
- SteamSpy використовував лише публічні дані, що обмежувало його надійність і повноту, оскільки він не міг враховувати індивідуальні профілі користувачів або точну кількість продажів.

Враховуючи дані фактори, важливо було використовувати SteamSpy з обережністю та не розглядати його дані як точну інформацію.

Також у якості аналога було розглянуто SuperData Research. SuperData Research - провідна аналітична компанія, що спеціалізується на зборі та аналізі даних у сфері відеоігор, мобільних додатків та інших цифрових розваг. Компанія заснована в 2009 році і пропонує широкий спектр послуг для розуміння ринку інтерактивних розваг.

У 2018 році SuperData Research була придбана компанією Nielsen, одним із світових лідерів у галузі вимірювання та аналітики аудиторії. Це стало важливим кроком у розвитку SuperData Research та розширенні її глобальної присутності.

SuperData Research є впливовою та авторитетною компанією в індустрії відеоігор та цифрових розваг, її дані та звіти регулярно цитуються у відомих

виданнях та засобах масової інформації, а її аналіз вважається важливим джерелом інформації для багатьох гравців ринку.

SuperData Research бере участь у конференціях та заходах, пов'язаних з індустрією відеоігор, представляючи результати досліджень та надаючи інформацію про ринкові тенденції. Вона також є активним учасником відомих виставок та заходів, таких як Electronic Entertainment Expo (E3) та Game Developers Conference (GDC).

SuperData Research збирає та аналізує дані про продажі відеоігор, мікротранзакції, активність гравців, доходи від мобільних додатків та інші показники галузі. Інформація збирається та аналізується за допомогою численних джерел даних, включаючи інтернет-магазини, соціальні мережі, розробників та видавців ігор.

Клієнтами SuperData Research є різноманітні суб'єкти індустрії відеоігор, зокрема розробники та видавці ігор, інвестори, маркетологи, виробники обладнання та інші зацікавлені сторони. Ми надаємо клієнтам дані та аналітичні звіти, які допомагають приймати рішення в різних аспектах їхнього бізнесу.

SuperData Research регулярно публікує новини, дослідження та звіти про ринок відеоігор та цифрових розваг. Ці матеріали містять найсвіжішу інформацію, тенденції та прогнози, які допоможуть вам зрозуміти ландшафт індустрії та розробити бізнес-стратегії.

SuperData Research пропонує платний план, який надає доступ до більш детальної та конкретної інформації. Це дозволяє клієнтам отримувати більш точні дані та аналіз для вирішення конкретних питань і проблем.

На рис. 1.2 можна побачити прогноз продаж гри, а саме розподіл ймовірності цифрових продаж. Продажі поділено між комп’ютером та консолею.

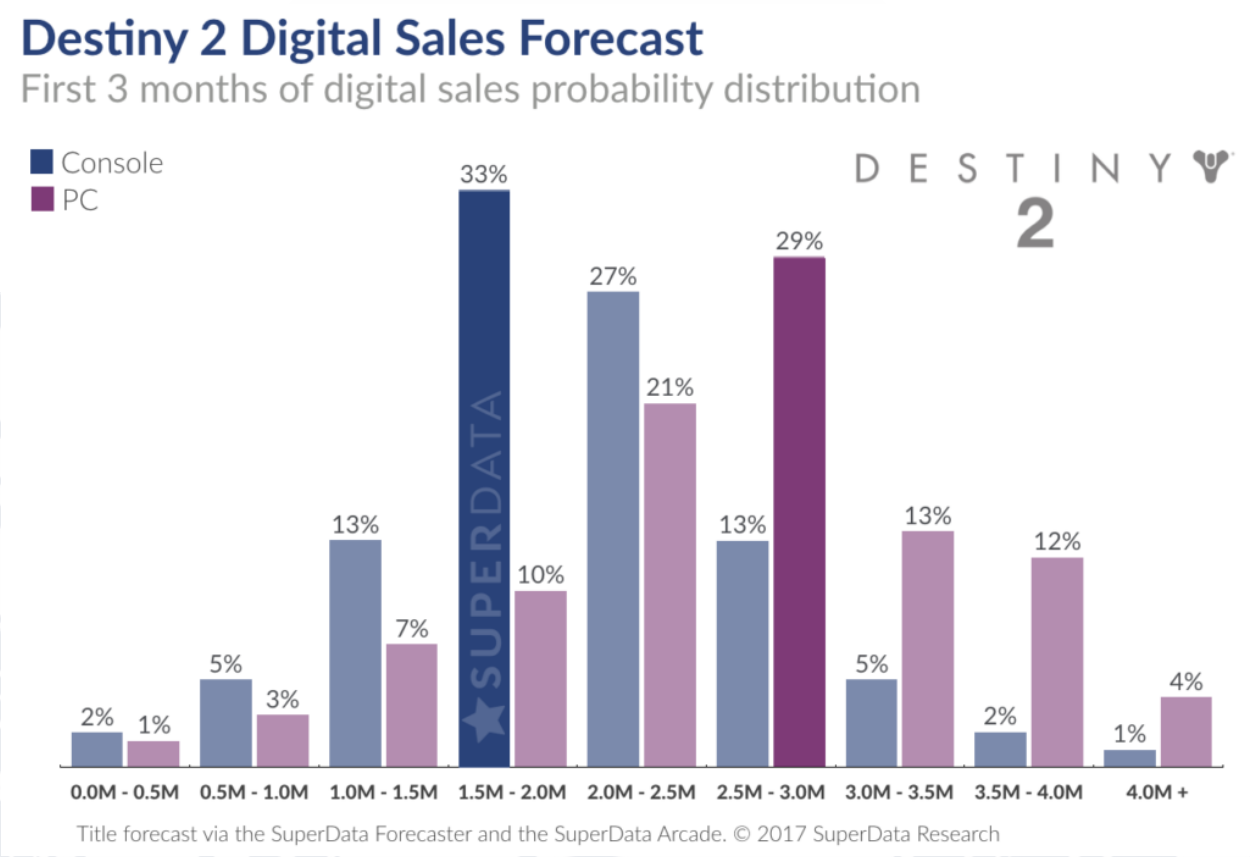

Рисунок 1.2 – Приклад оформлення прогнозу продажів відеоігор

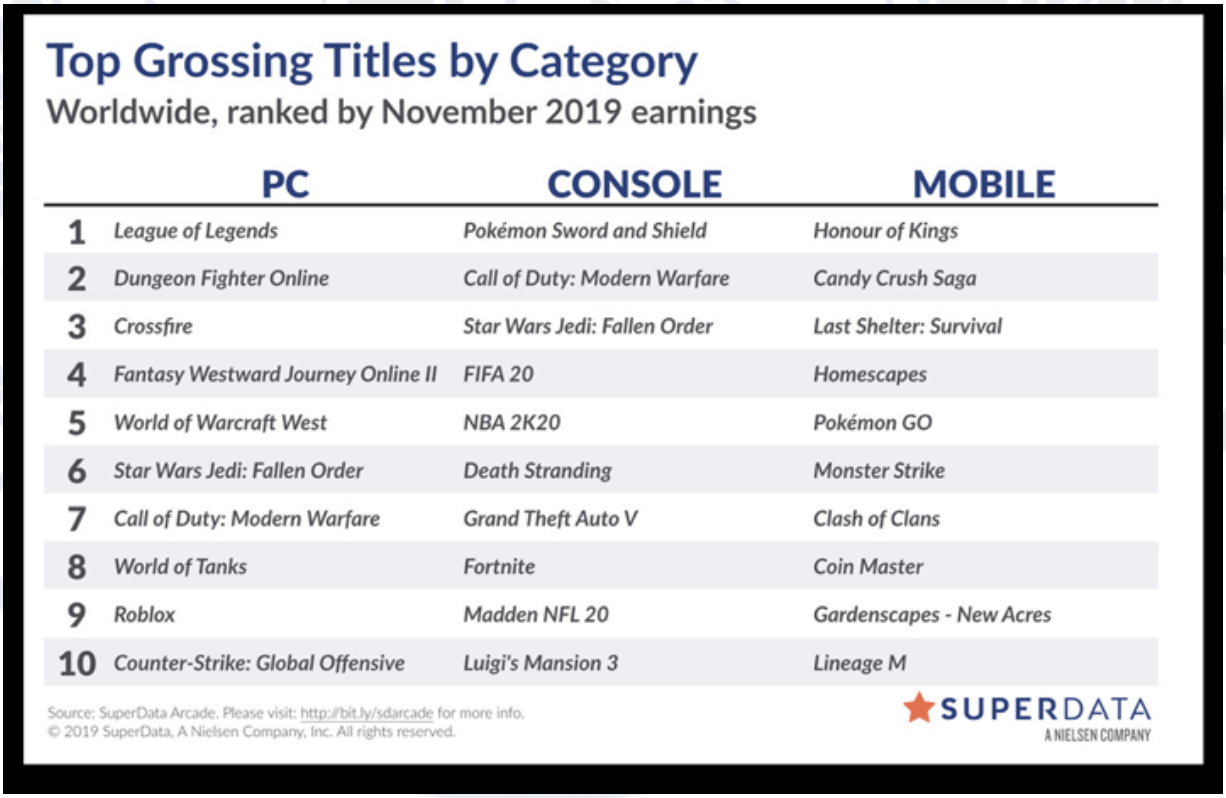

Рисунок 1.3 – Приклад оформлення звіту з відеоіграми, які є найбільш популярними у продажах на різних пристроях

Переваги використання SuperData Research:

- SuperData Research – визнаний лідер у сфері аналізу відеоігор та мобільних додатків. Маючи великий досвід у зборі та аналізі даних, компанія надає якісну інформацію та перспективи розвитку ринку розваг;
- SuperData Research збирає та аналізує широкий спектр даних, включаючи продажі ігор, доходи від мікротранзакцій, активність гравців та інші показники. Це дає цінну інформацію про ринок, конкуренцію та споживання цифрових розваг;
- інформація, надана SuperData Research, може бути використана для прийняття обґрунтованих рішень щодо стратегій розробки, маркетингу та дистрибуції відеоігор та мобільних додатків. Вона допомагає розробникам та видавцям ігор зрозуміти потреби ринку та відповідно сфокусувати свої зусилля.

Недоліки використання сервісу SuperData Research:

- повний доступ до послуг SuperData Research може бути досить дорогим - платні плани SuperData Research включають обмеження та плату за високий рівень деталізації та специфічні дані. Це може бути обмеженням для деяких користувачів з обмеженим бюджетом;
- індустрія відеоігор постійно розвивається, і дані, доступні від SuperData Research, можуть швидко застарівати. Це може бути проблемою для розробників і видавців, яким потрібна актуальна інформація для прийняття рішень;
- SuperData Research має видатну репутацію, але деякі критики ставлять під сумнів надійність і точність даних, частково через те, що методологію збору та аналізу даних важко перевірити.

1.3 Постановка задачі

Індустрія відеоігор-одна з найприбутковіших індустрій розваг, яка з року в рік зростає в плані продажів і прибутків. Розробляючи та просуваючи відеоігри на ринок, компанії стикаються з рішеннями, які можуть зробити їх успішними або зруйнувати їхні продажі.

Однак ринок відеоігор постійно змінюється, а смаки та вподобання гравців надзвичайно різноманітні. Тому розробникам і видавцям важливо мати доступ до методів аналізу даних, які допоможуть їм оцінити та спрогнозувати, наскільки добре продаватимуться їхні відеоігри.

Методи дослідження:

Для досягнення поставлених цілей використовуються такі методи:

- збір та обробка даних: збір релевантних даних про продажі програмного забезпечення для відеоігор, його характеристики та ринкові тенденції. Обробка даних та підготовка аналізу;
- статистичний аналіз: статистичні методи використовуються для виявлення кореляцій і тенденцій між різними факторами та продажами ігрового програмного забезпечення;
- машинне навчання: використання алгоритмів машинного навчання для побудови прогнозних моделей продажів відеоігор на основі наявних даних;
- експериментальний аналіз: порівняння та оцінка ефективності різних моделей і методів аналізу даних на основі реальних даних про продажі відеоігор;
- валідація результатів: тестування результатів на незалежних наборах даних для перевірки стабільності та точності моделей.

Очікується, що дослідження дасть такі результати:

- вивчення та порівняння різних методів аналізу даних для оцінки продажів відеоігор;

- розробка та оцінка ефективності моделей прогнозування продажів відеоігор на основі зібраних даних;
- висновки щодо найбільш ефективних методів та моделей оцінки продажів відеоігор;
- рекомендації розробникам та видавцям щодо використання аналітики даних для покращення стратегій маркетингу та розвитку відеоігор.

Для проведення дослідження використовувався датасет, що містив інформацію про продажі відеоігор, їх характеристики та ринкові тенденції.

Дані з датасета будуть систематизовані та піддані обробці для подальшого аналізу. Це включить в себе обчислення загальних продаж, визначення найкращої гри за певний період, а також аналіз трендів за роками. Також потрібно провести аналіз продажів відеоігор у різних регіонах, зокрема, визначення популярності ігор в Північній Америці, Європі та Азії, а також визначення найбільш популярних жанрів. Використати метод рекурентної нейронної мережі LSTM для прогнозування продаж відеоігор.

Як результат буде візуалізовано всі аспекти аналізу продажу відеоігор а також прогнозування продаж за допомогою рекурентної нейронної мережі LSTM.

Висновок до розділу

У даному розділі було розглянуто предметну галузь. Також показано аналоги, приведено їх переваги та недоліки. Та також була здійснена постановка задачі.

РОЗДІЛ 2
ВИКОРИСТАНІ ТЕХНОЛОГІЇ

2.1 Особливості Python

Python (найбільше вживане прочитання – «Пайтон») це інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією, розроблена Гвідован Россумом у 1990 році. Високорівневі структури даних разом із динамічною семантикою та динамічним зв'язуванням роблять її привабливою для швидкої розробки додатків і як засіб об'єднання існуючих компонентів. Python підтримує модулі та пакети модулів, що полегшує модуляризацію та повторне використання коду. Інтерпретатор і стандартна бібліотека Python доступні як у скомпільованому вигляді, так і у вигляді вихідних текстів на всіх основних платформах. Мова програмування Python підтримує декілька парадигм програмування [3].

Python відомий своїм чистим, зрозумілим синтаксисом, який є дружнім для початківців і легким для читання іншими розробниками.

Він використовує відступи для визначення блоків коду, тому в ньому немає фігурних дужок або інших громіздких символів для представлення блоків коду.

Python підтримує об'єктно-орієнтоване програмування, що дозволяє розробляти об'єктно-орієнтовані додатки, які сприяють структурованості та модульності коду.

Python автоматично визначає тип змінної на основі значення, яке вона містить, що дозволяє присвоювати значення змінній без зазначення її типу.

У Python прив'язка ідентифікаторів (змінних і функцій) до об'єктів відбувається під час виконання, а не під час компіляції.

Python має широкий набір стандартних бібліотек, включаючи модулі для роботи з файлами, мережами, базами даних, графікою, математичними операціями тощо.

Python підтримує модульну організацію коду. Ви можете створювати власні модулі та пакети для організації та структурування коду.

Python підтримує декілька парадигм програмування, включаючи об'єктно-орієнтоване, процедурне та функціональне програмування.

Python підтримується багатьма операційними системами, що дозволяє розробникам написати код один раз і запускати його на різних платформах без змін.

Python має велику та активну спільноту розробників, що робить її дуже популярною та гарантує швидкий розвиток і підтримку мови.

Python використовується для розробки веб-додатків, наукових обчислень, штучного інтелекту, аналізу даних, робототехніки, ігор, мережевого програмування та багатьох інших завдань.

Історія мови програмування Python дуже багата та доволі таки цікава. Розглянемо основні події та важливі моменти у розвитку даної мови [4]:

- Python створив голландський програміст Гідван Россум наприкінці1980-х років. Він розпочав роботу над мовою у грудні 1989 року;
- перша публічна версія Python (версія 0.9.0) була випущена в лютому 1991 року. Ця версія включала базові можливості мови, такі як цикли, умовні оператори, обробка винятків та функції;
- у 1994 році вийшла версія Python 1.0, яка включала в себе покращену систему модулів та багато інших нововведень;
- версія 2.0 Python була випущена у 2000 році, та мала важливі зміни, такі як підтримка Unicode, list comprehensions, garbage collection і багато іншого;
- у 2000 році було запроваджено PEP -систему пропозицій щодо розвитку мови, яка дозволяє розробникам співпрацювати і вносити ідеї та зміни до мови;
- Python 3.0 був великим релізом, що містив багато несумісних змін з Python 2.x. Його метою було полегшити розробку та виправити неоднозначності у синтаксисі Python;

- з часом Python продовжував розвиватися і став однією з найпопулярніших мов програмування у світі. Вона набула значної популярності в машинному навчанні, веб-розробці, аналізі даних та багатьох інших сферах;
- у 2018 році Гвідо ван Россум пішов у відставку з посади "Довічного диктатора-благодійника" (BDFL) і передав роль мовного менеджера іншому розробнику;
- у січні 2020 року офіційна підтримка Python 2.x закінчилася, і розробникам було запропоновано перейти на Python 3 для отримання підтримки та оновлень;
- у 2023 році серед IT-спеціалістів найпопулярнішою мовою програмування згідно рейтингу IEEE Spectrum стала Python. У даному рейтингу було налічено 59 мов програмування.

Так як Python було створено доволі таки пізно порівняно з іншими мовами програмування, через це Python була і є впливом багатьох інших мов програмування. Ось декілька із них [5]:

- R: Python конкурує з R в аналізі даних та статистиці, і багато бібліотек Python для аналізу даних натхненні ідеями та пакетами, наявними в R;
- JavaScript: JavaScript стає важливою мовою в контексті веб-розробки, а такі ініціативи, як Node.js, дозволяють запускати JavaScript на стороні сервера. Існують також такі проекти, як Brython, який дозволяє Python працювати в браузері;
- Java: у світі машинного навчання Python конкурує з Java. Це пов'язано з тим, що обидві мови виконують обчислення над великими обсягами даних;
- C: Python має вбудований C API, який може взаємодіяти з кодом, написаним на C. Це дозволяє розробникам використовувати ефективні бібліотеки, написані на C, з кодом на Python;

- Ruby: Ruby та Python мають багато спільного з точки зору читабельного та елегантного синтаксису, а також філософії "зручності та продуктивності розробника";
- Swift: деякі функції та ідеї, запроваджені у Swift, мові програмування для розробки додатків для iOS та macOS, також були використані у проектах Python для мобільних платформ.

Дані взаємодії та впливи роблять мову програмування Python більш універсальною та потужною, а також сприяють поширенню кращих практик та інновацій у світі програмування.

Python є потужною мовою програмування з численними перевагами, але також вона має деякі обмеження. Тому варто вказати дані переваги та недоліки [6].

Переваги:

- простий, зрозумілий синтаксис: Python відомий своєю чистою, зрозумілою структурою синтаксису, що робить його ідеальним вибором як для початківців, так і для досвідчених користувачів;
- велика стандартна бібліотека: Python надає безліч модулів і бібліотек для різноманітних завдань, спрощуючи розробку і дозволяючи швидко створювати функціональні додатки;
- кросплатформенність: Python підтримується багатьма операційними системами і не залежить від платформи;
- велика спільнота та підтримка: Python має активну та дружню спільноту розробників, яка пропонує безкоштовну допомогу та безліч ресурсів для вивчення та вдосконалення мови;
- мультипарадигмальність: Python підтримує багато парадигм програмування, включаючи об'єктно-орієнтоване, процедурне та функціональне програмування;

- розширюваність: API C/C++ дозволяє легко взаємодіяти з кодом, написаним іншими мовами програмування, також можна використовувати бібліотеки інших мов програмування;
- актуальність для наукових досліджень та аналізу даних: Python є популярним інструментом в обробці даних, машинному навчанні, наукових обчисленнях та широкому спектрі галузей;
- розробка веб-додатків: Python має популярні фреймворки для розробки веб-додатків, такі як Django та Flask.

Недоліки Python:

- низька швидкість виконання: Python є інтерпретованою мовою, що може призвести до повільнішого виконання великих обчислень та операцій, що вимагають високої швидкості, порівняно з скомпільованими мовами;
- обмеження в мобільній розробці: хоча Python підтримує мобільну розробку (наприклад, через фреймворк Kivy), він не такий популярний, як Java або Kotlin у цій сфері;
- обсяг пам'яті: Python може вимагати більше пам'яті, ніж інші мови програмування;
- глобальний випуск нових версій: перехід на нову версію Python може бути складним, оскільки іноді вимагає внесення змін до існуючого коду;
- глобальне блокування інтерпретатора (GIL): Python має GIL, який обмежує паралельну обробку на багатопроцесорних системах, що може бути проблемою при одночасній обробці багатьох завдань.

Не зважаючи на дані недоліки, Python такій залишається однією з найбільш популярних та важливих мов програмування у світі завдяки своїм перевагам та різностороннім застосуванням.

Фреймворки та бібліотеки:

- NumPy та SciPy: Використовуються для наукових обчислень та роботи з числовими даними;
- Pandas: Для роботи з даними та аналізу;

- Django та Flask: Популярні веб-фреймворки для розробки веб-додатків.

Машинне навчання:

- scikit-learn: Бібліотека для класичного машинного навчання;
- TensorFlow та PyTorch: Фреймворки для глибокого навчання.

Візуалізація даних:

- Matplotlib та Seaborn: Для створення графіків та візуалізації даних.

Робота з базами даних:

- SQLAlchemy: ORM (Object-Relational Mapping) для роботи з реляційними базами даних;
- MongoEngine: Для роботи з MongoDB.

Асинхронне програмування:

- asyncio: вбудований модуль для асинхронного програмування.

Розробка ігор:

- Pygame: бібліотека для створення ігор.

Конкурентність та паралельність:

- multiprocessing: для паралельного програмування;
- concurrent.futures: для асинхронного програмування.

Модулі для роботи з мережами:

- Requests: для взаємодії з веб-серверами;
- socket: для низькорівневої роботи з мережами.

Python має декілька версій які є актуальними, розглянемо деталі стосовно Python 2 та Python 3:

- Python 2: Перша версія була випущена у 2000 році і довгий час була основною версією мови. Однак офіційна підтримка Python 2 припинилася у 2020 році. Багато бібліотек та фреймворків перейшли на підтримку Python 3;
- Python 3: Випущений у 2008 році, Python 3 має на меті виправити помилки дизайну в Python 2 та вдосконалити мову; Python 3 включає

синтаксичні та структурні зміни для написання більш чистого та ефективного коду.

Різниця між Python 2 та Python 3:

- синтаксис: Python 3 внесе деякі зміни в синтаксис, такі як використання дужок для функції print і зміни у роботі з рядками;
- юнікод: Python 3 використовує Unicode як основний тип для рядків, що полегшує роботу з різними мовами;
- функції: деякі функції та бібліотеки були змінені чи перейменовані;
- друк: у Python 3 print - це функція, а не оператор, як у Python 2;
- ділення: у Python 3 результат ділення цілих чисел повертає дійсне число, щоб уникнути непорозумінь.

Перехід на Python 3:

- багато бібліотек і проектів активно підтримують Python 3, але деякі ще не підтримують його повністю;
- для полегшення переходу існують інструменти, такі як 2 to 3, які автоматично конвертують код з Python 2 на Python 3.

Підтримка Python 2:

- після припинення офіційної підтримки Python 2 рекомендується уникати використання цієї версії у нових проектах;
- для додатків, які все ще використовують Python 2, рекомендується розглянути можливість переходу на Python 3, щоб забезпечити довгострокову підтримку.

Майбутнє Python:

- Python продовжує розвиватися, регулярно випускаються нові версії; документація PEP та обговорення у спільноті допомагають визначити майбутній напрямок розвитку мови.

У Python також є інструменти для розповсюдження Python-програм, особливо коли важко встановити Python на кінцевих комп’ютерах або коли ви

хочете забезпечити простоту використання для кінцевого користувача. Тому варто розглянути дані інструменти більш детально.

Cython - це мова програмування, яка розширює функціональність Python, дозволяючи використовувати функції C в Python - коді.

Cython дозволяє писати код, подібний до Python, але додає можливість вказувати типи для змінних, що покращує продуктивність.

Використовують Cython для того щоб розширити Python-код для отримання кращої швидкодії та ефективності, а також для того щоб викликати функції із коду, написаного на мові C, прямо в Python.

PyInstaller - це інструмент для створення автономних виконуваних файлів (exe) з програм на Python.

PyInstaller пакує код Python, інтерпретатор Python та залежності в один виконуваний файл, що полегшує розповсюдження програмних продуктів. Використовують для розповсюдження Python-додатків як самостійних виконуваних файлів.

cx_Freeze – ще один інструмент для створення автономних виконуваних файлів з Python-програм. cx_Freeze має більше опцій конфігурації, що дозволяє більш точно налаштувати процес збирання. Аналогічно, використовується для створення виконуваних файлів з Python-коду.

Обидва ці інструменти не потребують встановлення Python на комп'ютер користувача. Вони дозволяють користувачам створювати автономні додатки, які можна легко використовувати без необхідності встановлювати інтерпретатор Python або інші залежності. Ці інструменти корисні для розповсюдження програм на Python, особливо коли важко встановити Python на кінцевих точках або коли кінцеві користувачі хочуть зробити його простим у використанні.

PEP (Python Enhancement Proposals) - це механізм, який використовується для опису нових можливостей, поліпшень і змін у мові програмування Python. PEP – документи створюються спільнотою розробників Python і слугують основою для обговорення та прийняття рішень

щодо розвитку мови. Вони слугують основою для обговорення та прийняття рішень щодо розвитку мови [7].

PEP структура:

- номер: кожен PEP має унікальний номер для полегшення ідентифікації в індексі PEP;
- назва: ідентифікує запропоновану зміну або інновацію;
- автор: ім'я та контактна інформація автора(ів) PEP;
- статус: вказує на статус PEP (прийнято, відхилено, призупинено тощо);
- опис: короткий опис та мета PEP;
- деталі: містить детальну інформацію про технічні аспекти та принципи PEP;
- рекомендації: вказує, як слід впроваджувати PEP і як слід контролювати прийняття PEP.

Типи PEP:

- PEP стандартів які описують нові функції чи зміни в мові;
- PEP процесу які вносять зміни в сам процес розробки Python;
- PEP інформаційні які надають загальну інформацію або керівництво.

Прийняття PEP вимагає достатньої підтримки з боку спільноти та позитивного рішення керівного органу, такого як Python Steering Council. Обговорення та огляди відбуваються на відкритих форумах, таких як python-dev, а також на різних онлайн-платформах для спільної розробки коду (наприклад, GitHub). Розвиток Python виконується ітеративно через прийняття та реалізацію різних PEP. Прозорість розробки сприяє взаємодії розробників та спільноти під час обговорення та прийняття рішень.

Приклади PEP:

- PEP 20 (The Zen of Python): Загальні принципи дизайну для Python;

- PEP 8: Стиль програмування для мови Python;
- PEP 484: Аннотації типів для Python (введено в Python 3.5);
- PEP 333 (Python Web Server Gateway Interface): визначає стандарти сумісності між веб-серверами та програмами на Python, що важливо для впровадження веб-серверів та фреймворків;
- PEP 572 (Assignment Expressions): Введення нового синтаксису для присвоювання значень змінним виразами (оператор ‘:=’), що робить код більш зрозумілим та компактним.

Документи PEP визначають ключові аспекти розвитку мови та задають курс на прийняття стандартів та інновацій у світі Python. Таким чином, це важливий інструмент для організації та спрощення процесу прийняття рішень у спільноті розробників Python.

Завдяки своїй простоті, гнучкості та розширюваності Python використовується у великих проектах та у різних галузях. Ось кілька прикладів масштабних проектів, де Python виявився дуже ефективним:

- Dropbox використовує Python як одну з основних мов програмування для розробки свого серверного програмного забезпечення. Це включає в себе велику кількість back-end та системні компоненти;
- один з найпопулярніших сервісів соціальних мереж, Instagram, також розроблений на Python; Django, популярний веб-фреймворк для Python, використовується для створення та підтримки багатьох частин інфраструктури Instagram;
- Pinterest – ще один великий веб-сервіс, який використовує Python для своєї інфраструктури; Django також використовується для створення та підтримки додатку Pinterest;
- YouTube використовує Python для частини своєї інфраструктури. Зокрема, існує фреймворк Tornado, який використовується для обробки запитів у режимі реального часу.

У цих проектах Python використовується для розробки веб-сайтів, обробки даних, створення API, реалізації бек-ендів та інших завдань. Зручний синтаксис, численні бібліотеки та фреймворки, а також активна спільнота роблять її привабливим варіантом для великих і складних проектів.

Отож як висновок можна сказати, що мова програмування Python є найпопулярніших, впливових і універсальних мов у світі розробки програмного забезпечення. Її успіх базується на простому, зрозумілому синтаксисі, багатій стандартній бібліотеці, мультипарадигмальній природі та активній спільноті розробників.

Python використовується в різних галузях, включаючи веб-розробку, наукові дослідження, аналіз даних, машинне навчання, робототехніку іт. д. Роль Python в розробці і його вплив на інші мови і технології є важливим для індустрії програмування.

Python залишається важливим інструментом для розробників, і його майбутнє обіцяє подальші інновації та розвиток. Таким чином, Python є важливою мовою програмування для різноманітних завдань і продовжує набирати популярність у світі інформаційних технологій.

2.2 Бібліотеки обробки даних

У роботі використовуються наступні бібліотеки для аналізу та візуалізації даних: pandas, matplotlib.pyplot, numPy, seaborn, plotly.express, plotly.graph_objects а також keras. Ці бібліотеки дозволяють виконувати різноманітні операції з даними, створювати графіки та візуалізації для аналізу даних про продажі відеоігор.

Pandas – це бібліотека програмного забезпечення, написана для мови програмування Python для управління та аналізу даних. Зокрема, він надає структури даних та операції для управління числовими таблицями та часовими рядами. Pandas - це вільне програмне забезпечення, випущене під триточковою ліцензією BSD. Назва походить від терміна "панельні дані", який

відноситься до багатовимірних структурованих наборів даних в економетриці[7].

Основними функціями бібліотеки Pandas є:

Фрейм даних і серія: Pandas визначає 2 основні типи даних: фрейм даних і серія. Фрейм даних - це двовимірна таблиця з даними, подібними до аркуша Excel, тоді як серія – це тривимірна структура даних, подібна до стовпця або масиву в таблиці.

Завантаження та зберігання даних: Pandas дозволяє легко завантажувати дані з різних джерел, включаючи CSV, Excel, бази даних SQL та JSON. Ви також можете зберігати свої дані в різних форматах.

Індексація та вибір даних: Pandas надає зручний інструмент для вибору та редагування фреймів даних та рядів даних за допомогою індексів та міток.

Обробка та очищення даних: Бібліотека дозволяє виконувати різні операції з обробки даних, включаючи фільтрацію, сортування, групування, агрегування та видалення дублікатів.

Візуалізація даних: Pandas працює з іншими бібліотеками візуалізації, такими як Matplotlib та Seaborn, для створення графіків та діаграм на основі ваших даних.

Робота з часовими рядами: у Pandas є спеціальний інструмент для роботи з часовими рядами для аналізу тимчасових даних.

Інтеграція з іншими бібліотеками: Pandas легко інтегрується з іншими популярними бібліотеками, такими як NumPy, Scikit-Learn і т.д.

Pandas спрощує і підвищує ефективність обробки і аналізу даних на Python завдяки своїм потужним функціям і зручному інтерфейсу. Він широко використовується в аналізі даних, наукових дослідженнях, фінансах, техніці та інших галузях.

Pandas використовується для різних завдань, пов'язаних з обробкою, аналізом та маніпулюванням даними. Ось деякі основні операції та завдання, які можна виконувати за допомогою Pandas:

- зчитувати дані з різних форматів, таких як CSV, Excel, SQL, JSON, HTML і т.д. Pandas також надає можливість зберігати дані в різних форматах;
- створювати DataFrame та Series, об'єднувати, розділювати, видаляти та змінювати стовпці;
- вибірка та індексація даних за допомогою міток індексів, рядків та стовпців;
- обробляти відсутні значення, фільтрація дублікатів, перейменування стовпців і інші операції очищення даних;
- групування даних відповідно до певних критеріїв і виконування функції агрегування;
- створювання графіків та діаграм, використовуючи інтеграцію Matplotlib.

Ці функції в Pandas роблять його потужним інструментом для маніпулювання та вивчення ваших даних. Бібліотека є основою для аналізу даних у Python і використовується в різних галузях, таких як наука про дані, фінанси, біологія та інженерія.

Pandas має свої переваги та недоліки, тому варто розглянути їх.

Переваги Pandas:

- Pandas має 2 зручні структури даних: DataFrame і Series, що робить роботу з даними простою і ефективною;
- здатність ефективно обробляти великі набори даних і обробляти з використанням різнорідних типів даних;
- він підтримує ряд функцій для маніпулювання даними, включаючи групування, агрегування, обробку пропущених значень, фільтрацію та сортування;
- зручний спосіб читання і запису даних в різних форматах, таких як CSV, Excel, SQL, JSON, HTML і т. д;

- інтеграція з Matplotlib дозволяє виводити дані безпосередньо з Pandas в графіки і діаграми;
- Pandas широко використовується в науці даних, фінансах, біології, інженерії і інших галузях;
- підтримка великої та активної спільноти користувачів, а також детальна та проста для розуміння документація.

Також Pandas має кілька недоліків:

- Pandas можуть використовувати значну кількість пам'яті, особливо для великих наборів даних. Це може призвести до обмежень при роботі з великими обсягами даних;
- деякі операції Pandas займають багато часу в порівнянні з іншими оптимізованими бібліотеками, такими як NumPy, особливо при роботі з великими даними;
- Pandas специфічний для мови програмування Python і може бути обмежений, якщо вам потрібно інтегруватися з кодом, написаним іншими мовами;
- деякі операції та функції можуть бути важкими для освоєння новачками, особливо тим, хто немає досвіду обробки даних.

Незважаючи на ці недоліки, Pandas є однією з найпопулярніших і використовуваних бібліотек для обробки та аналізу даних у середовищах Python.

Matplotlib – це бібліотека мови програмування Python для візуалізації даних у 2D-графіці (також підтримується 3d-графіка). Отримане зображення може бути використано в якості ілюстрації в публікації [8].

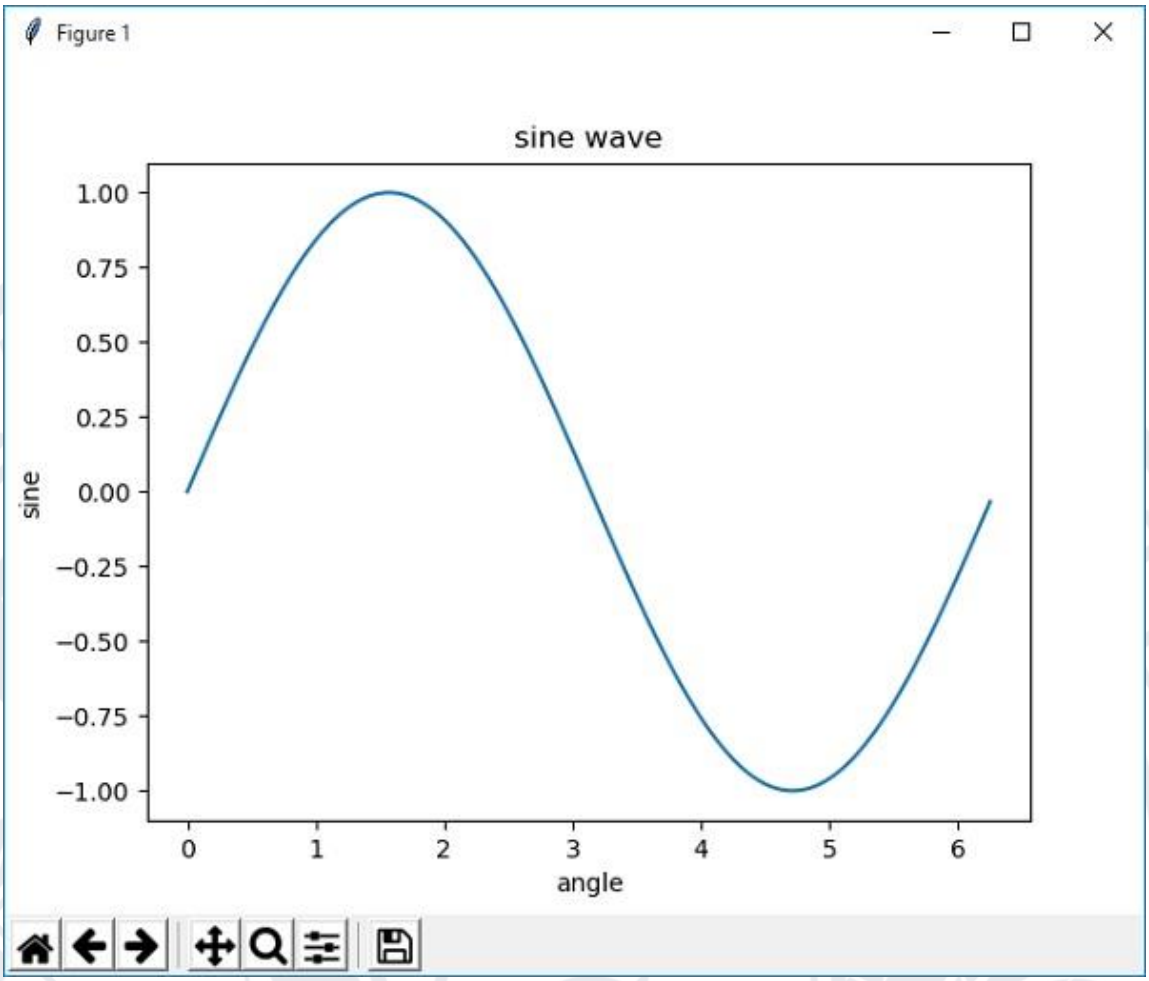

Рисунок 2.1 – Приклад 2д графіка

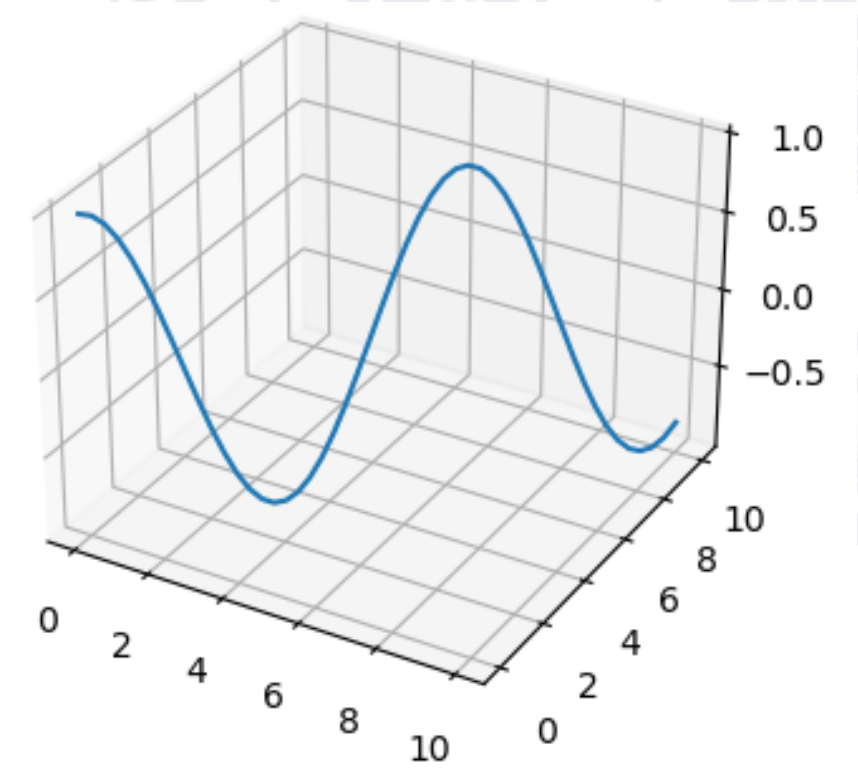

Рисунок 2.2 – Приклад 3д графіка

На рисунку 2.1 та рисунку 2.2 можна побачити приклад зображення 2d та 3d графіка, а саме у даному випадку це синусоїда, де вісь У це синус а вісь Х є кутом.

Matplotlib – одна з найбільш широко використовуваних бібліотек візуалізації даних у галузі наукових обчислень, аналізу даних та машинного навчання. Це дозволяє користувачам створювати зображення та графіки з розширеним налаштуванням та контролем відображення даних.

Matplotlib має свої переваги та недоліки, розглянемо їх.

Переваги Matplotlib:

- Matplotlib – це універсальна бібліотека для створення різних графіків та діаграм різних типів даних;
- простий та інтуїтивно зрозумілий інтерфейс дозволяє користувачам з різним рівнем досвіду отримувати доступ до Matplotlib;
- Matplotlib підтримує безліч різних графічних функцій, дозволяючи користувачам створювати різні типи візуалізацій;
- бібліотека широко використовується в наукових дослідженнях, інженерії, аналізі даних, машинному навчанні і інших галузях;
- можливість створення тривимірних графіків дозволяє візуалізувати дані в тривимірному просторі;
- Matplotlib можна використовувати для створення інтерактивних діаграм у Jupyter Notebook інших інтерактивних середовищах;
- він має велику активну спільноту користувачів, а також детальну документацію та приклади використання.

Недоліки Matplotlib:

- може знадобитися багато коду та параметрів конфігурації, щоб досягти ідеального вигляду графіка;
- деякі API Matplotlib можуть виглядати застарілими порівняно з сучасними бібліотеками візуалізації, такими як Seaborn;

- стиль програмування Matplotlib може бути менш ефективним і більш рутинним порівняно з іншими бібліотеками, такими як Seaborn і Plotly;
- для деяких початківців синтаксис Matplotlib здається неінтуїтивним, особливо якщо порівнювати його з іншими бібліотеками візуалізації;
- Matplotlib немає графічного редактора, тому він може спростити визначення зовнішнього вигляду графіка для тих, хто не хоче або не може маніпулювати кодом;
- інтерактивні функції Matplotlib, хоча і присутні, менш розвинені в порівнянні з іншими бібліотеками, такими як Plotly.

Незважаючи на ці недоліки, Matplotlib є однією з найпопулярніших і широко використовуваних бібліотек для візуалізації даних у середовищах Python. Її міццю і гнучкістю користуються мільйони дослідників і розробників по всьому світу.

Ключові функції бібліотеки Matplotlib включають [9]:

Створення різних типів діаграм: можна створювати лінійні, точкові, стовпчасті, кругові, 3D та інші типи графічних візуалізацій.

Налаштування діаграми: надається повний контроль над зовнішнім виглядом діаграми, включаючи зміну кольорів, стилів ліній, маркерів, умовних позначень, масштабування та інших параметрів.

Підтримка підписів і маркерів: можливість додавати підписи до графіків, ідентифікувати точки даних і створювати умовні позначення для опису графічних об'єктів.

Збереження графіків: є можливість зберігати свої графіки в різних форматах, включаючи PNG, JPEG, PDF і SVG.

Інтерактивність: Matplotlib підтримує можливість створення інтерактивних графіків, з якими можна взаємодіяти, наприклад, масштабувати, перетягувати та переглядати додаткові дані.

Бібліотека Matplotlib – це потужний інструмент візуалізації даних Python, що дозволяє аналітикам та дослідникам візуалізувати та аналізувати дані та приймати обґрунтовані рішення [10].

Бібліотека Seaborn використовується для візуалізації даних з моделей, створених на основі наборів даних, прогнозування результатів та аналізу мінливості даних [11].

Seaborn спрощує процес створення графіків і діаграм, дозволяючи користувачам висловлювати складні статистичні взаємозв'язки в більш зрозумілій формі. Він пропонує стилізовану та готову до використання кольорову палітру, а також функції для автоматичного обчислення та відображення статистичних показників, таких як середнє значення, дисперсія, кореляція та інші.

Ключові особливості бібліотеки Seaborn включають:

Спеціальні функції для візуалізації даних: Seaborn можна використовувати для побудови різних типів графіків, таких як лінійчасті графіки, стовпчасті гістограми, прямокутні графіки, графіки розсіювання, теплові карти і т.д.

Стиль і колірна палітра: Seaborn має вбудований стиль і колірну палітру, які є стильними і привабливими, що дозволяє легко створювати красиву графіку.

Підтримка статистичного аналізу: бібліотека може виконувати Статистичний аналіз даних і відображати результати на графіках, таких як карти розподілу, діаграми взаємодії пар, графіки регресії і багато інших.

Інтерфейс високого рівня: Seaborn забезпечує простий та інтуїтивно зрозумілий інтерфейс для створення графіків, що дозволяє швидко та ефективно візуалізувати дані без необхідності використання великої кількості коду.

Seaborn – це корисна бібліотека для аналізу та візуалізації даних на Python, яка часто використовується спільно з іншими бібліотеками, такими як Pandas та Matplotlib, для створення корисних візуалізацій даних.

Бібліотека Seaborn надає багато функцій для візуалізації даних за допомогою різних графіків. Ось певний тип графіків, який зазвичай використовується в Seaborn:

- гістограми (у Seaborn sns.histplot). Гістограма використовується для відображення розподілу одновимірних даних. Розбиває діапазон значень на комірки (стовпці), щоб вказати кількість випадкових подій у кожній комірці(приклад гістограми зображено на рисунку 2.3);

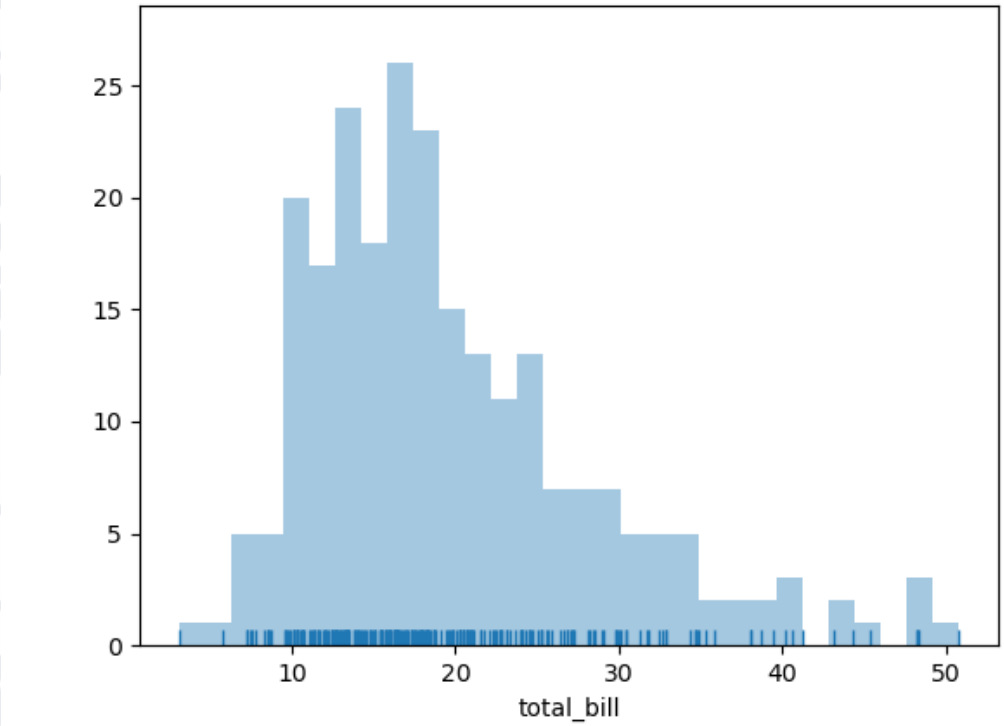

Рисунок 2.3 – Гістограма Seaborn

- діаграма розсіювання (у Seaborn sns.scatterplot). Діаграма розсіювання використовується для відображення взаємодії між двома змінними. Кожна точка представляє пару значень, і їх положення на графіку може вказувати на ступінь кореляції (приклад діаграми розсіювання можна побачити на рисунку 2.4, на даному графіку зображено по вісь У ефективність палива а по вісі Х показано об’єм двигуна);

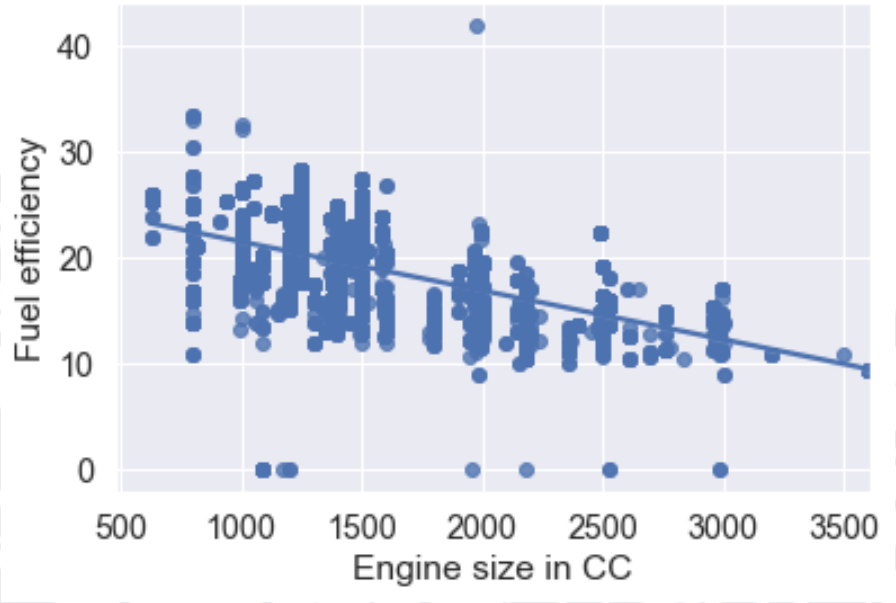

Рисунок 2.4 – Діаграма розсіювання Seaborn

- діаграма взаємодії пар (у Seaborn sns.pairplot). Використовується для відображення взаємодії між усіма парами змінних у наборі даних (приклад діаграми взаємодії пар показано на рисунку 2.5);

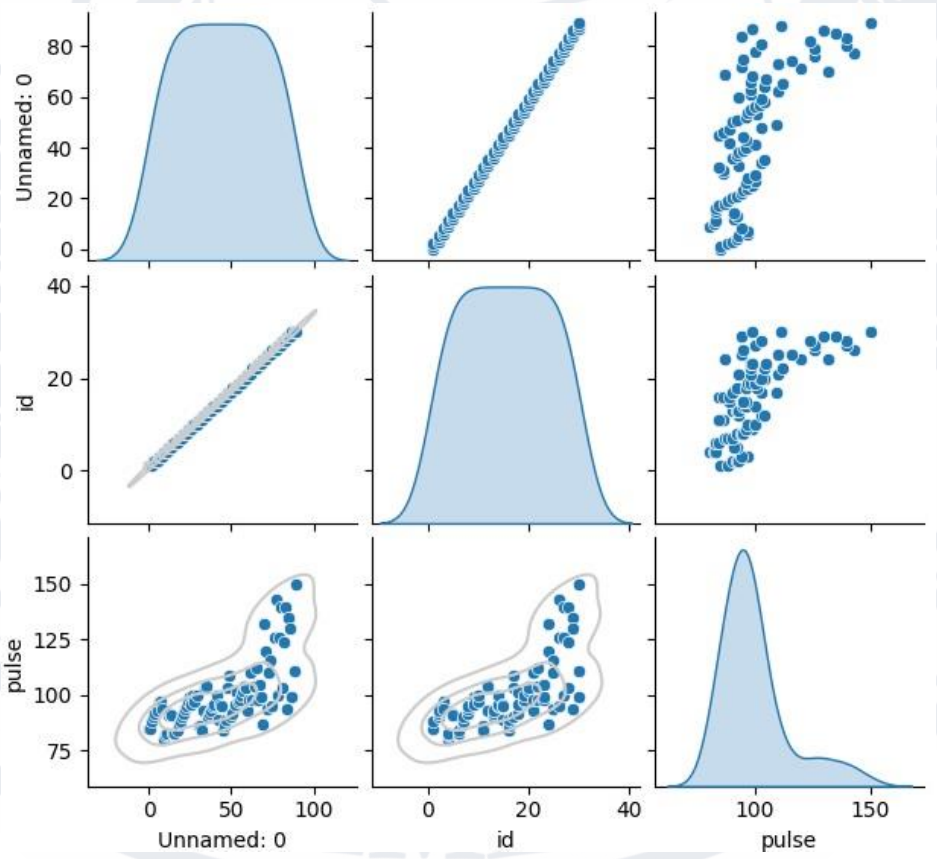

Рисунок 2.5 – Діаграма взаємодії пар Seaborn

- теплові карти (у Seaborn sns.heatmap). Їх використовують для відображення матриць даних, в яких кольори представляють величину значень (приклад теплової карти зображено на рисунку 2.6).

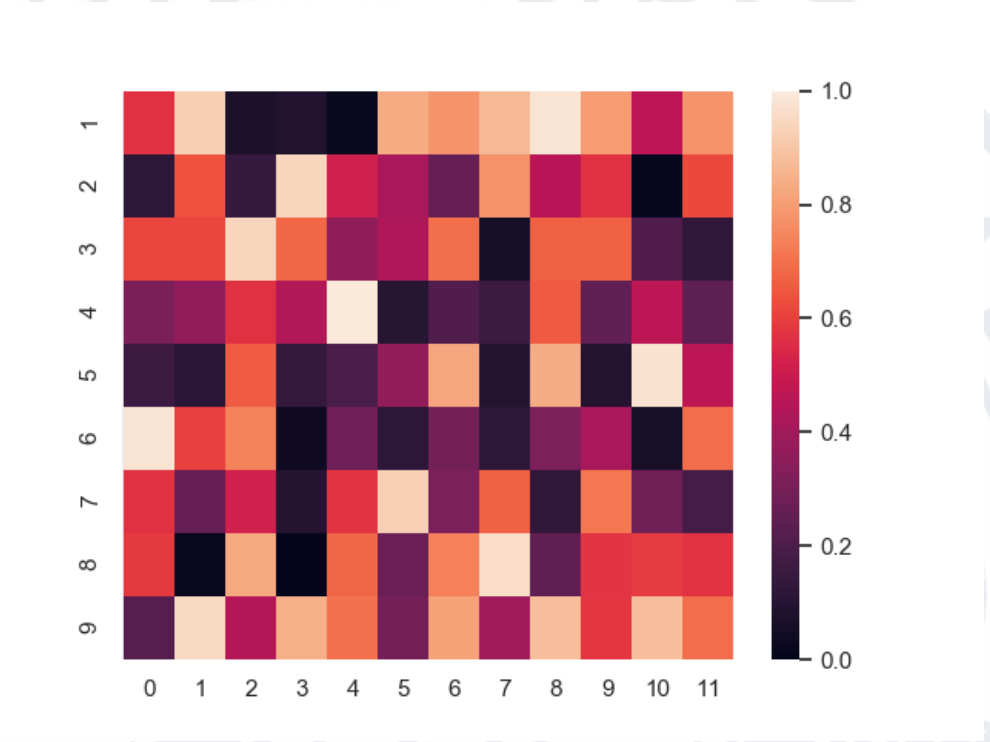

Рисунок 2.6 - Теплова карта Seaborn

Seaborn має свої переваги та недоліки, які варто розглянути. Отож переваги Seaborn:

- у Seaborn простий та інтуїтивно зрозумілий синтаксис, що робить його зручним у використанні, особливо для початківців;
- Seaborn легко інтегрується з бібліотекою Pandas, полегшуючи візуалізацію даних з фреймів даних Pandas;
- бібліотека має гарний стиль і вбудовану колірну палітру, що дозволяє створювати привабливу і чітку графіку без необхідності в багатьох налаштуваннях;
- Seaborn надає API високого рівня для візуалізації статистичних залежностей від ваших даних, що робить його простим у використанні та розумінні;

- бібліотека надає функції для візуалізації статистичних взаємодій, таких як кореляція, регресія та розподіл;
- Seaborn – це досить гнучка бібліотека, яку ви можете легко масштабувати та налаштовувати відповідно до власних потреб.

Недоліки:

- одне із обмеження Seaborn полягає в тому, що воно може бути менш гнучким порівняно з більшими бібліотеками, такими як Matplotlib. Це може статися, якщо вам потрібно реалізувати складний тип діаграми або нестандартний тип діаграми;
- деякі типи діаграм та налаштування можуть бути складними або не підтримуваними в Seaborn. У таких випадках вам буде потрібно об'єднувати їх з Matplotlib;
- Seaborn ідеально підходить для роботи зі структурованими даними. Для деяких неструктурованих або складних даних може знадобитися використання додаткових бібліотек або обробників;
- Seaborn побудований на Matplotlib, тому вам потрібно використовувати Matplotlib безпосередньо, щоб використовувати більш вдосконалені та нестандартні функції;
- якщо вам потрібно виконати складні перетворення або обробку даних перед рендерингом, вам слід використовувати інші бібліотеки, такі як Pandas або NumPy.

Важливо зазначити, що вибір Seaborn та інших бібліотек залежить від конкретних потреб та уподобань. Seaborn часто використовується як зручний інструмент для швидкої та ефективної візуалізації статистики.

Plotly – це бібліотека для створення інтерактивних візуалізацій даних мовою програмування Python та іншими мовами програмування. Він надає інструменти для створення різних типів діаграм, включаючи лінійні діаграми, стовпчасті діаграми, кругові діаграми, діаграми розсіювання, 3D-діаграми, теплові карти, географічні карти та багато інших [12].

Основними особливостями Plotly є:

Інтерактивність: Plotly дозволяє створювати інтерактивні діаграми, які можна масштабувати, переміщувати, виділяти та маніпулювати мишею, приклад інтерактивного графіка можна побачити на рисунку 2.7. Це робить графік більш зручним для аналізу даних та перегляду результатів.

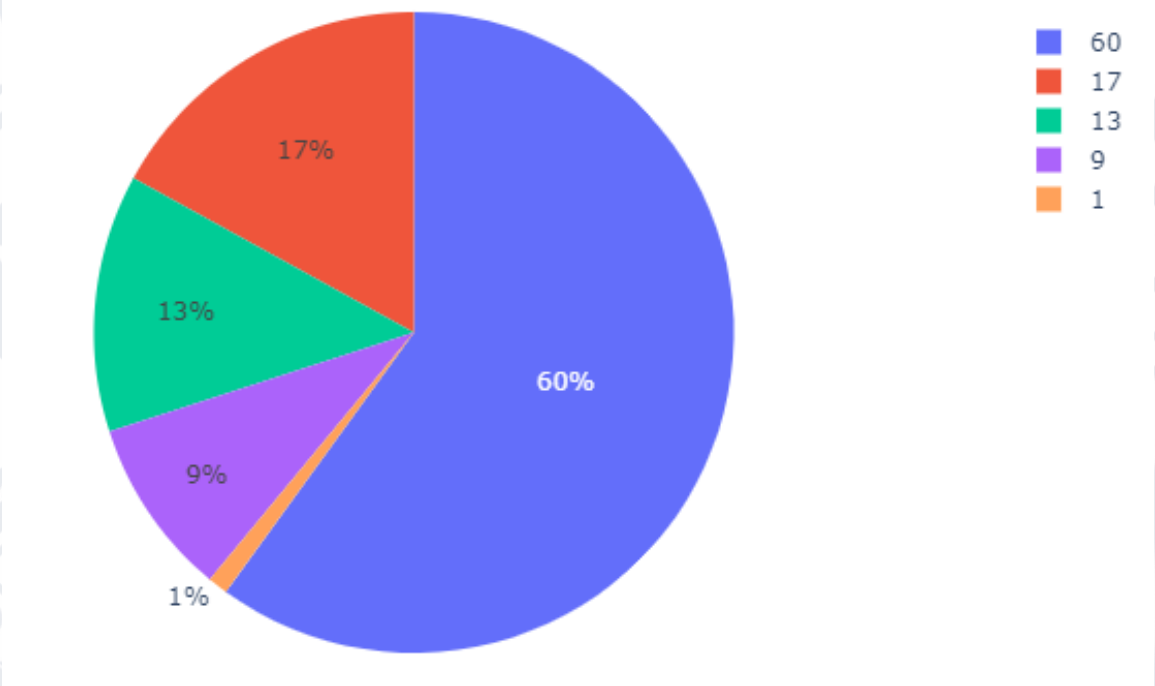

Рисунок 2.7 – Приклад діаграми з використанням Plotly

Підтримка багатьох мов програмування: Plotly можна використовувати з такими мовами програмування, як Python, R, Julia і MATLAB, так що ви можете легко створювати візуалізації на ваших улюблених мовах.

Високоякісна візуалізація: графіки, створені в Plotly, мають високий рівень якості і можуть бути використані для презентацій і публікацій.

Підтримка онлайн – сервісів Plotly: plotly надає онлайн-сервіс для зберігання і публікації візуалізацій, що дозволяє вам ділитися графіками з іншими користувачами і вбудовувати їх на веб-сторінки.

Підтримка різних типів діаграм: Plotly підтримує створення різних типів діаграм (приклад візуалізації діаграми можна побачити на рисунку 2.8), від простих лінійних діаграм до складних 3D (приклад 3D графіка можна побачити на рисунку 2.9) і географічних візуалізацій.

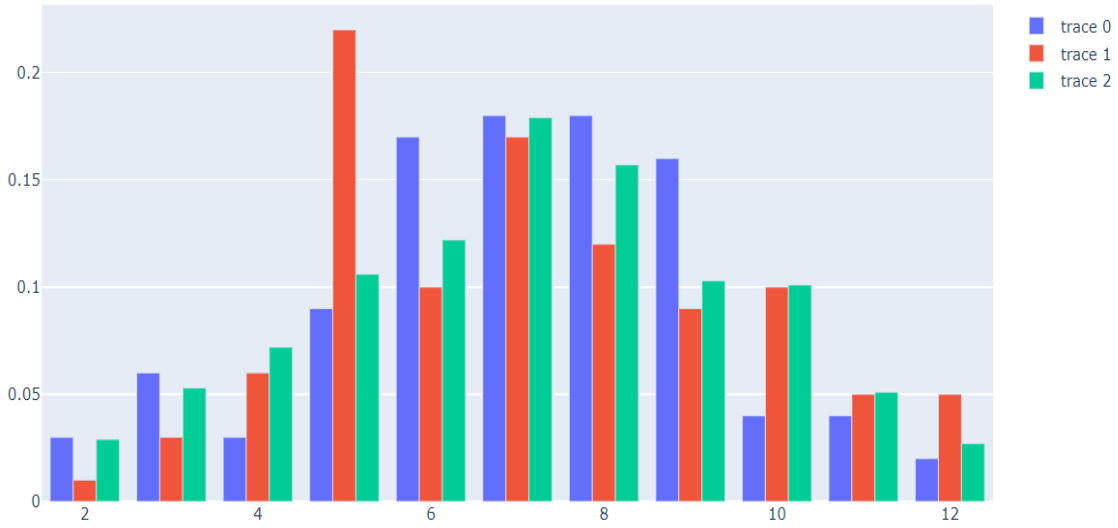

Рисунок 2.8 – Приклад візуалізації діаграм

Загалом, Plotly – це потужний інструмент для створення інтерактивних та високоякісних візуалізацій даних у різних сферах, таких як аналіз даних, наукові дослідження, бізнес-аналіз та веб-розробка.

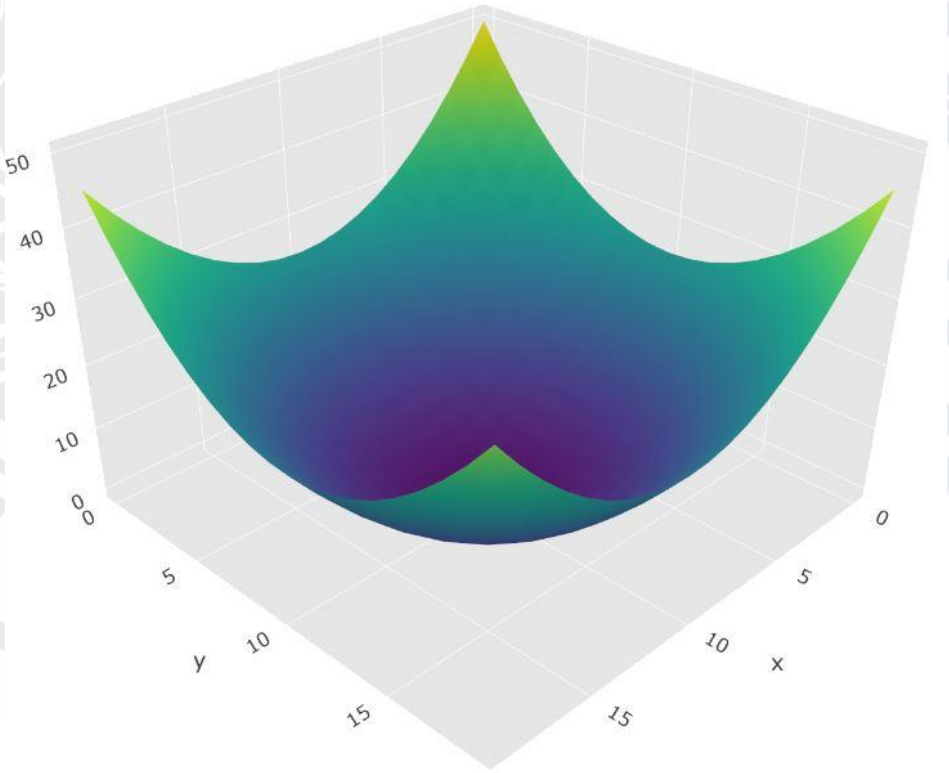

Рисунок 2.9 – Приклад візуалізації 3D графіка

Plotly як бібліотека має свої переваги та недоліки які варто розглянути. Отож переваги Plotly:

- одною із головних переваг Plotly є інтерактивність графіка. Користувачі можуть наблизити зображення, збільшити масштаб, перемістити та отримати детальну інформацію, що дозволяє їм ефективно вивчати дані;
- Plotly підтримує різні типи діаграм, включаючи стовпчасті діаграми, гістограмні діаграми, кругові діаграми, теплові карти та 3D – діаграми;
- Plotly має простий та інтуїтивно зрозумілий синтаксис. Крім того, Plotly Express надає інтерфейс високого рівня для швидкого створення графіків у спрощеному коді;
- Dash, фреймворк розроблений Plotly, дає змогу створювати інтерактивні веб-додатки для того щоб виконувати аналіз даних, інтегруючи графіки безпосередньо в веб-додаток.

Plotly також має недоліки, а саме:

- при великих обсягах даних, особливо при великій кількості точок на графіку, обробка і відображення великої кількості точок може знизити ефективність інтерактивних діаграм;
- безкоштовна версія Plotly має обмеження, які можуть включати обмеження кількості запитань на сторінці та способи збереження та експорту графіків;
- інтерактивні діаграми Plotly засновані на онлайн-сервісі для створення та відображення Plotly, тому для їх повноцінного використання часто потрібне підключення до інтернету.

Plotly це потужний інструмент для створення інтерактивних та високоякісних візуалізацій даних у Python і також інших мовах програмування. Його переваги містять у собі інтерактивність графіків, різноманітність типів діаграм, простий синтаксис і підтримку багатьох мов.

Незважаючи на обмеження в роботі з великими обсягами даних і обмеження в безкоштовній версії, Plotly є і буде популярним інструментом для візуалізації даних та аналізу.

NumPy (Numerical Python) - це бібліотека для мови програмування Python, яка підтримує великі багатовимірні масиви та матриці, а також великий набір функцій для маніпулювання цими масивами. NumPy - основна бібліотека для наукових обчислень у середовищі Python [13].

Особливостями NumPy є:

- NumPy надає numpy.ndarray, що представляє собою багатовимірний масив даних одного типу. Це дозволяє ефективно виконувати операції з великими обсягами даних;
- NumPy містить ряд математичних функцій для виконання різних операцій, включаючи лінійну алгебру, тригонометричні функції та функції зворотного виклику;
- NumPy надає різні способи доступу до елементів масиву та виконання операцій з фрагментами;
- модуль NumPy має функції для генерації випадкових чисел;
- Broadcasting – це механізм, який дозволяє NumPy виконувати операції з масивами різних розмірів і форм, полегшуючи кодування та дозволяючи явне програмування.

NumPy є важливим інструментом для багатьох галузей, таких як наука про дані, машинне навчання, обробка зображень, моделювання та багато інших. Потрібно вказати переваги та недоліки даної бібліотеки.

Переваги даної бібліотеки:

- операції NumPy реалізовані на C, тому вони швидші порівняно зі стандартними структурами даних Python, особливо при роботі з великими обсягами даних;
- масиви NumPy мають однаковий тип даних, що полегшує та прискорює обчислення;

- NumPy надає ряд функцій для виконання математичних, статистичних операцій та операцій лінійної алгебри;
- механізм broadcasting дозволяє виконувати операції з масивами різної форми і розміру. Це полегшує кодування та зменшує необхідність явного повторення коду;
- можливості індексації і нарізки NumPy спрощують доступ до підмасивів і їх зміни.

Також NumPy має свої недоліки:

- NumPy може використовувати багато пам'яті, особливо для великих масивів. У деяких випадках це може призвести до неефективного використання ресурсів;
- NumPy дуже популярний у спільноті вчених та інженерів які використовують Python, але він специфічний для інших мов програмування. Це може бути проблемою, якщо вам потрібно використовувати його іншими мовами програмування;
- новачкам може бути важко освоїти всі функції та правила NumPy, особливо якщо немає великого досвід у програмуванні;
- У деяких аспектах можливості обробки та аналізу даних у NumPy поступаються бібліотекам високого рівня, таким як pandas.

Незважаючи на ці недоліки, NumPy є невід'ємною частиною екосистеми Python для наукових обчислень і пропонує багато переваг для роботи з числовими даними та масивами.

NumPy пропонує широкий спектр можливостей для маніпулювання числовими даними та масивами мовою програмування Python. Розглянемо кілька прикладів того, що можна створити та запустити в NumPy:

- створення одновимірних та багатовимірних масивів:

```
# Одновимірний масив
arr1d = np.array([1, 2, 3, 4, 5])
# Багатовимірний масив
```

```
arr2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
```

- виконання арифметичних операцій з масивами:

```
arr = np.array([1, 2, 3, 4, 5])
# Додавання константи
arr_plus_2 = arr + 2
# Множення на константу
arr_times_3 = arr * 3
```

- виконання операцій лінійної алгебри, такі як обчислення зворотні матриці чи власних значень:

```
matrix = np.array([[1, 2], [3, 4]])
# Обернена матриця
inverse_matrix = np.linalg.inv(matrix)
# Власні значення та власні вектори
eigenvalues, eigenvectors = np.linalg.eig(matrix)
```

- генерація випадкових чисел та створення послідовностей чисел:

```
# Випадкові числа
random_numbers = np.random.rand(3, 3)
# Послідовність чисел
sequence = np.arange(0, 10, 2)
```

- звертання до елементів масиву та використання зрізів:

```
arr = np.array([1, 2, 3, 4, 5])
# Звертання до елементу
element = arr[2]
# Зріз
subarray = arr[1:4]
```

- використання NumPy у поєднанні з іншими бібліотеками (наприклад, pandas) для обробки та аналізу даних.

Бібліотека NumPy та інші бібліотеки в екосистемі Python дозволяють виконувати потужні обчислення, маніпулювати числовими даними та створювати графіки. Однак важливо зазначити, що потрібно буде

маніпулювати графіками та візуалізацією даних, використовуючи додаткові бібліотеки, такі як Matplotlib та Seaborn. Загальна потужність і гнучкість NumPy дозволяє створювати складні обчислення і аналіз числових даних на мові програмування Python.

Бібліотека Keras – це інструмент нейронної мережі високого рівня, заснований на різних бібліотеках низького рівня для обчислень, таких як tensorflow, Theano та Microsoft Cognitive Toolkit (CNTK). Keras 2.3.0 та новіші версії є офіційними бібліотеками високого рівня, які полегшують створення, навчання та використання нейронних мереж. Він включений до TensorFlow як API рівня [14].

Особливості Keras:

- Keras дає простий та інтуїтивно зрозумілий інтерфейс для створення нейронних мереж. Можна легко визначати шари, компілювати моделі та навчати їх за допомогою декількох рядків коду;
- моделі Keras створюються з використанням шарів. Ви можете легко додавати, видаляти або змінювати шари для налаштування архітектури вашої мережі;
- Keras має широкий спектр вбудованих шарів для різних типів робіт, включаючи повністю підключені шари, збудження, зменшення, повторювані шари (такі як LSTM, GRU) та багато інших;
- починаючи з версії 2.3.0, Keras став офіційним API високого рівня для tensorflow. Тепер ви можете використовувати Keras з TensorFlow і насолоджуватися всіма перевагами обох бібліотек;
- Keras може використовувати TensorFlow, Theano і CNTK як бекенд для того щоб виконувати обчисленя.

Також Keras має свої недоліки. Інтеграція Keras з TensorFlow полегшує його використання, але в особливо складних випадках ви можете втратити гнучкість у порівнянні з чистим TensorFlow. Keras став API високого рівня для tensorflow, тому до деяких специфічних для TensorFlow функцій важко отримати доступ або для їх реалізації потрібно багато коду. Для досліджень,

які потребують глибокої настройки або використання незвичайних шарів, вам може знадобитися доступ низько горівня, який keras надає не в повній мірі. У деяких випадках Keras може бути обмеженням з точки зору архітектурної свободи, особливо при роботі з нетрадиційними мережевими архітектурами.

Загалом, використання Keras є розумним вибором для тих, хто цінує швидкість розробки, простоту використання та хоче ефективно використовувати можливості нейронних мереж у своїх проектах, не глибоко занурюючись у деталі реалізації. Keras є привабливим варіантом, особливо для початківців, дослідників та інженерів, які шукають зручний інтерфейс для швидкого розгортання моделей.

Висновки до розділу

У розділі було розглянуто основний функціонал мови програмування Python, переваги та недоліки даної мови.

Також було розглянуто бібліотеки Python, які використовувались у роботі а саме: pandas, matplotlib.pyplot, numPy, seaborn, plotly.express, plotly.graph_objects а також keras. Було розглянуто переваги та недоліки даних бібліотек а також наведено приклади робіт даних бібліотек

РОЗДІЛ 3
АНАЛІЗ ДАНИХ ДЛЯ ОЦІНКИ ПРОДАЖУ ВІДЕОІГОР

3.1 Аналіз датасета

У роботі проведено аналіз даних про продажі відеоігор, а також створено візуалізацію для кращого розуміння цих даних.

На рис. 3.1 наведено датасет Video Games Sales, з якого була отримана інформація. В даному датасеті є така інформація як: ім’я гри, платформа на якій випускалась гра, рік виготовлення, жанр, а також є ще декілька стовпців, які не наведено на рисунку, а саме: видавець ігор, продажі у північній Америці, продажі у Європі, продажі у Японії та інші продажі. Даний датасет містить у собі 11563 ігор.

Name	Platform		Year_of_Release		Genre	
11563 unique values	PS2	13%	2008	9%	Action	20%
	DS	13%	2009	9%	Sports	14%
	Other (12406)	74%	Other (13866)	83%	Other (11001)	66%
Wii Sports	Wii		2006		Sports	
Super Mario Bros.	NES		1985		Platform	
Mario Kart Wii	Wii		2008		Racing	
Wii Sports Resort	Wii		2009		Sports	
Pokemon Red/Pokemon Blue	GB		1996		Role-Playing	
Tetris	GB		1989		Puzzle	
New Super Mario Bros.	DS		2006		Platform	
Wii Play	Wii		2006		Misc	
New Super Mario Bros. Wii	Wii		2009		Platform	
Duck Hunt	NES		1984		Shooter	
Nintendogs	DS		2005		Simulation	

Рисунок 3.1 – Датасет Video Games Sales

Загальна структура програми аналізу продажу відеоігор, яка наведена на рис. 3.2 включає в себе етапи завантаження даних, їх обробку та аналіз, обчислення статистики, вибір найкращих ігор а також візуалізацію даних.

Рисунок 3.2 – Загальна структура програми аналізу продажу відеоігор

Для початку були підключені всі необхідні бібліотеки та завантажено датасети за допомогою бібліотеки «pandas» із файлів CSV, які містять інформацію про продажі відеоігор. Після цього було виведено перші рядки кожного завантаженого датасету, щоб переглянути, як виглядають дані.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go

dataset1 = pd.read_csv("C:\\Users\\Vlad\\OneDrive\\Робочий стіл\\Video_sales\\Video_Games_Sales_as_at_22_Dec_2016.csv")
dataset2 = pd.read_csv("C:\\Users\\Vlad\\OneDrive\\Робочий стіл\\Video_sales\\PS4_GamesSales.csv", encoding='latin-1')
dataset3 = pd.read_csv("C:\\Users\\Vlad\\OneDrive\\Робочий стіл\\Video_sales\\XboxOne_GameSales.csv", encoding='latin-1')
```

```
print(dataset1.head())
print(dataset2.head())
print(dataset3.head())
```

```
========== RESTART: C:/Users/Vlad/OneDrive/Робочий стіл/my-diplom.py ==========
                     Name Platform  ...  Developer Rating
0               Wii Sports      Wii  ...   Nintendo      E
1         Super Mario Bros.      NES  ...         NaN    NaN
2            Mario Kart Wii      Wii  ...   Nintendo      E
3         Wii Sports Resort      Wii  ...   Nintendo      E
4  Pokemon Red/Pokemon Blue       GB  ...         NaN    NaN

[5 rows x 16 columns]
                       Game    Year  ... Rest of World Global
0         Grand Theft Auto V  2014.0  ...          3.02  19.39
1  Call of Duty: Black Ops 3  2015.0  ...          2.44  15.09
2      Red Dead Redemption 2  2018.0  ...          2.26  13.94
3         Call of Duty: WWII  2017.0  ...          2.12  13.40
4                  FIFA 18    2017.0  ...          1.73  11.80

[5 rows x 9 columns]
   Pos                      Game    Year  ... Japan Rest of World  Global
0    1        Grand Theft Auto V  2014.0  ...  0.01          0.76    8.72
1    2  Call of Duty: Black Ops 3  2015.0  ...  0.02          0.68    7.37
2    3        Call of Duty: WWII  2017.0  ...  0.00          0.57    6.23
3    4     Red Dead Redemption 2  2018.0  ...  0.00          0.54    5.77
4    5                  MineCraft  2014.0  ...  0.00          0.49    5.43
```

Рисунок 3.3 – Перші рядки датасетів

Наступним кроком було виведено описову статистику для кожного з трьох завантажених наборів даних про продажі відеоігор. «Dataset Stats» виводить заголовок набору даних. «dataset.describe» генерує описову статистику для наборів даних, включаючи середнє значення (mean), стандартне відхилення (std), мінімальне (min) та максимальне (max) значення, а також квартилі (25%, 50%, 75%) для числових стовпців цього набору даних. Це допомагає отримати уявлення про характеристики даних у кожному наборі і підготувати їх до подальшого аналізу.

```
print("Dataset 1 Stats:")
print(dataset1.describe())
print("Dataset 2 Stats:")
print(dataset2.describe())
print("Dataset 3 Stats:")
```

```
print(dataset3.describe())
```

```
Dataset 1 Stats:
       Year_of_Release      NA_Sales  ...   User_Score     User_Count
count     16450.000000  16719.000000  ...  7590.000000   7590.000000
mean       2006.487356      0.263330  ...     7.125046    162.229908
std           5.878995      0.813514  ...     1.500006    561.282326
min        1980.000000      0.000000  ...     0.000000      4.000000
25%        2003.000000      0.000000  ...     6.400000     10.000000
50%        2007.000000      0.080000  ...     7.500000     24.000000
75%        2010.000000      0.240000  ...     8.200000     81.000000
max        2020.000000     41.360000  ...     9.700000  10665.000000

[8 rows x 10 columns]
Dataset 2 Stats:
              Year  North America  ...  Rest of World       Global
count   825.000000    1034.000000  ...    1034.000000  1034.000000
mean   2015.966061       0.204613  ...       0.089014     0.576054
std       1.298360       0.563471  ...       0.249410     1.583534
min    2013.000000       0.000000  ...       0.000000     0.000000
25%    2015.000000       0.000000  ...       0.000000     0.000000
50%    2016.000000       0.020000  ...       0.010000     0.060000
75%    2017.000000       0.120000  ...       0.050000     0.357500
max    2020.000000       6.180000  ...       3.020000    19.390000

[8 rows x 6 columns]
Dataset 3 Stats:
              Pos         Year  ...  Rest of World      Global
count  613.000000   505.000000  ...     613.000000  613.000000
mean   307.000000  2015.821782  ...       0.039951    0.438874
std    177.102136     1.382975  ...       0.089343    0.983147
min      1.000000  2013.000000  ...       0.000000    0.000000
25%    154.000000  2015.000000  ...       0.000000    0.000000
50%    307.000000  2016.000000  ...       0.010000    0.060000
75%    460.000000  2017.000000  ...       0.030000    0.380000
max    613.000000  2020.000000  ...       0.760000    8.720000
```

Рисунок 3.4 – Описова статистика

У цій частині виконувались наступні дії:

- Обчислюються загальні продажі для першого набору даних dataset1. Вибирається стовпець "Global_Sales", і метод .sum() обчислює суму всіх значень в цьому стовпці. Результат (сума продажів) зберігається у змінній total_sales1.
- Знаходиться найкраща гра за продажами у першому наборі даних dataset1. dataset1['Global_Sales'].idxmax() визначає індекс рядка, де "Global_Sales" має найбільше значення (найкращий продаж). Потім

dataset1.loc[...] використовує цей індекс, щоб отримати весь рядок з даними про цю найкращу гру. Результат зберігається у змінній best_selling_game1.

- Інформація у best_selling_game1 включає наступні стовпці, як "Name" (назва гри), "Platform" (платформа), "Year_of_Release" (рік виходу), "Genre" (жанр), "Publisher" (видавець), "NA_Sales" (продажі в Північній Америці), "EU_Sales" (продажі в Європі), "JP_Sales" (продажі в Японії), "Other_Sales" (продажі в інших регіонах), і "Global_Sales" (глобальні продажі), а також деякі інші дані, такі як "Critic_Score," "User_Score," "User_Count," "Developer," і "Rating."

Ці дані представляють інформацію про загальні продажі для першого набору даних, а також інформацію про найбільш продавану гру у цьому наборі даних, включаючи статистику продажів у різних регіонах та деяку інформацію про відгуки та рейтинги цієї гри.

```
total_sales1 = dataset1['Global_Sales'].sum()
print(f'Загальні продажі (Dataset 1): {total_sales1} мільйонів копій')
best_selling_game1 = dataset1.loc[dataset1['Global_Sales'].idxmax()]
print(f'Найкраща гра за продажами (Dataset 1):')
print(best_selling_game1)
```

```
Загальні продажі (Dataset 1): 8920.300000000001 мільйонів копій
Найкраща гра за продажами (Dataset 1):
Name               Wii Sports
Platform                  Wii
Year_of_Release        2006.0
Genre                  Sports
Publisher            Nintendo
NA_Sales                41.36
EU_Sales                28.96
JP_Sales                 3.77
Other_Sales              8.45
Global_Sales            82.53
Critic_Score             76.0
Critic_Count             51.0
User_Score                8.0
User_Count              322.0
Developer            Nintendo
Rating                      E
Name: 0, dtype: object
```

Рисунок 3.5 – Загальні продажі та найкраща гра

Далі за допомогою «sns.set(style="whitegrid")» встановлено стиль для графіків засобами бібліотеки Seaborn. У нашому випадку стиль вказує, що графіки будуть мати білий фон з сіткою.

Створюємо лінійний графік (line plot) для глобальних продажів відеоігор (Global_Sales) у залежності від року випуску гри (Year_of_Release). Використовує дані з dataset1. Отже наступний код створює лінійний графік, який демонструє динаміку глобальних продажів відеоігор протягом років.

```
sns.set(style="whitegrid")
plt.figure(figsize=(12, 6))
sns.lineplot(x='Year_of_Release', y='Global_Sales', data=dataset1, ci=None)
plt.title('Глобальні продажі відеоігор за роками')
plt.xlabel('Рік')
plt.ylabel('Глобальні продажі (в мільйонах копій)')
plt.xticks(rotation=45)
plt.show()
```

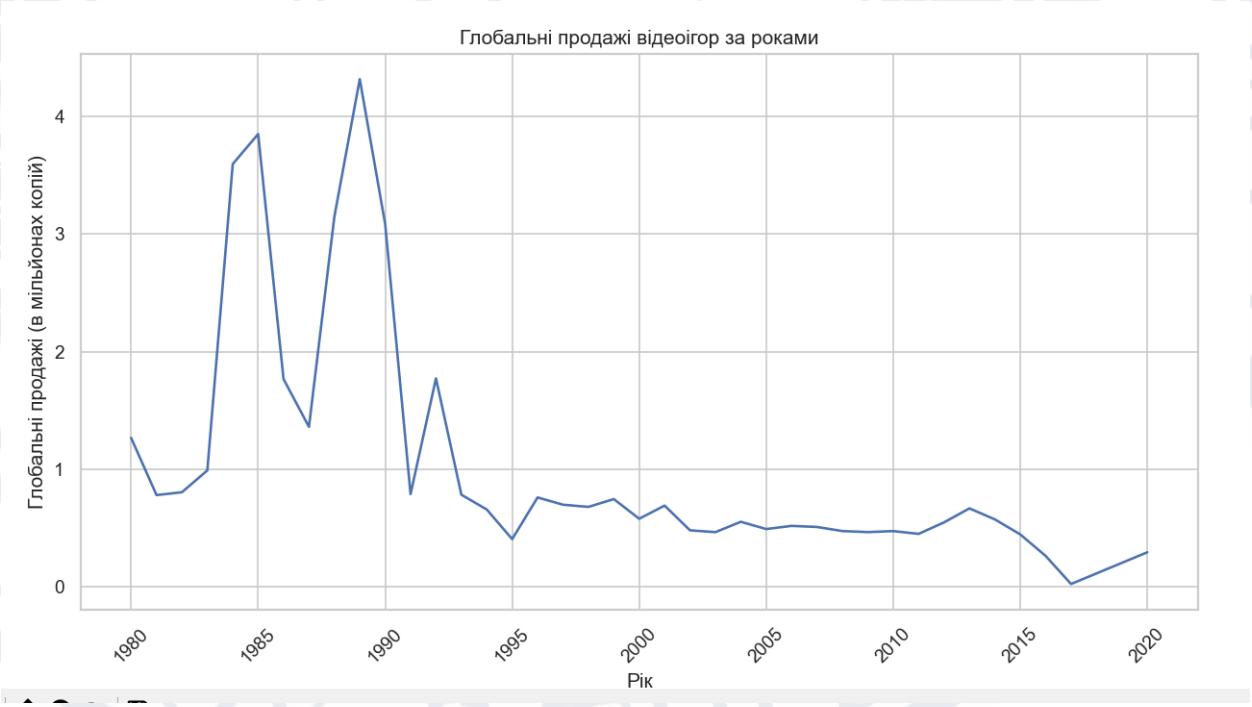

Рисунок 3.6 – Глобальні продажі відеоігор за роками

Наступним кроком виконується візуалізація графіку стовпчастої діаграми для глобальних продажів відеоігор за жанрами у другому наборі

даних ‘dataset2’. Створюється стовпчаста діаграма, де по осі «x» відображаються жанри відеоігор (Genre), а по осі «y» - глобальні продажі відеоігор (Global) у мільйонах копій.

```
plt.figure(figsize=(12, 6))
sns.barplot(x='Genre', y='Global', data=dataset2, ci=None)
plt.title('Глобальні продажі відеоігор за жанрами (Dataset 2)')
plt.xlabel('Жанр')
plt.ylabel('Глобальні продажі (в мільйонах копій)')
plt.xticks(rotation=90)
plt.show()
```

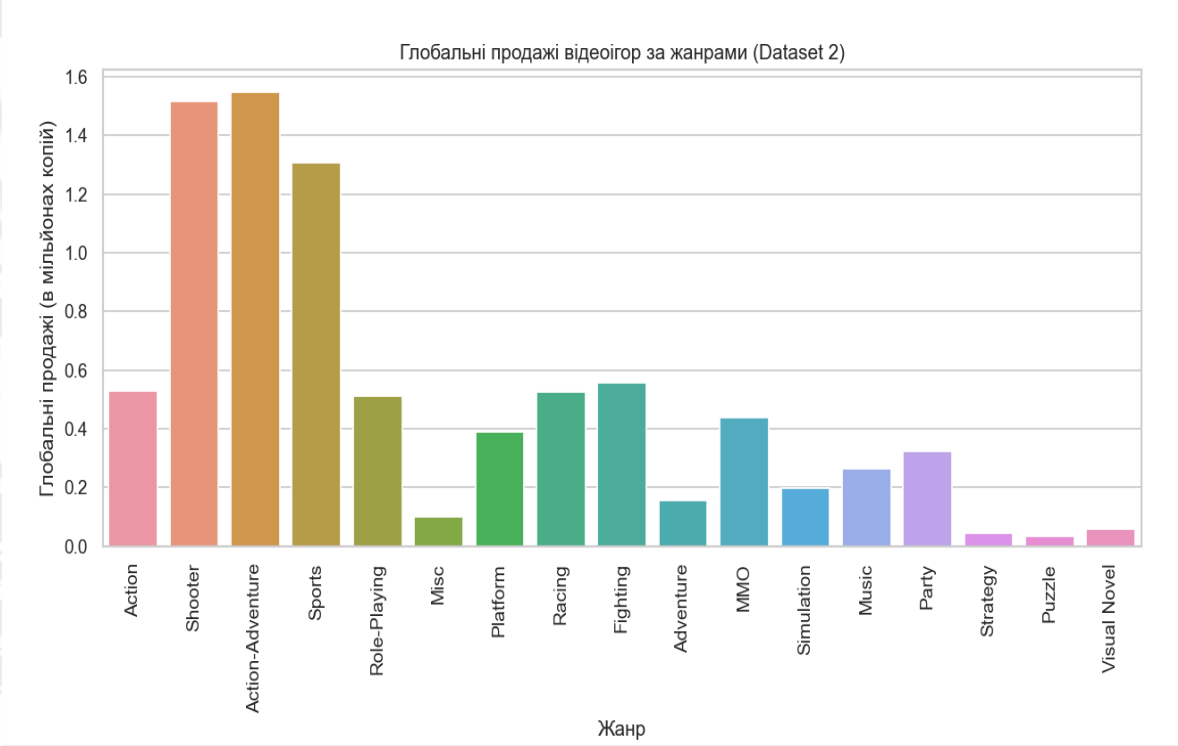

Рисунок 3.7 – Глобальні продажі за жанрами

Наступний фрагмент коду виконує візуалізацію стовпчастої діаграми (bar plot) для глобальних продажів відеоігор за платформами у третьому наборі даних ‘dataset3’.

Створює стовпчикову діаграму, де по осі «x» відображаються назви видавців відеоігор (Publisher), а по осі «y» - глобальні продажі цих відеоігор (Global) у мільйонах копій.

```
 plt.figure(figsize=(12, 6))
sns.barplot(x='Publisher', y='Global', data=dataset3.head(10), ci=None)
plt.title('Глобальні продажі відеоігор за платформами (перші 10) (Dataset 3)')
plt.xlabel('Платформа')
plt.ylabel('Глобальні продажі (в мільйонах копій)')
plt.xticks(rotation=45)
plt.show()
```

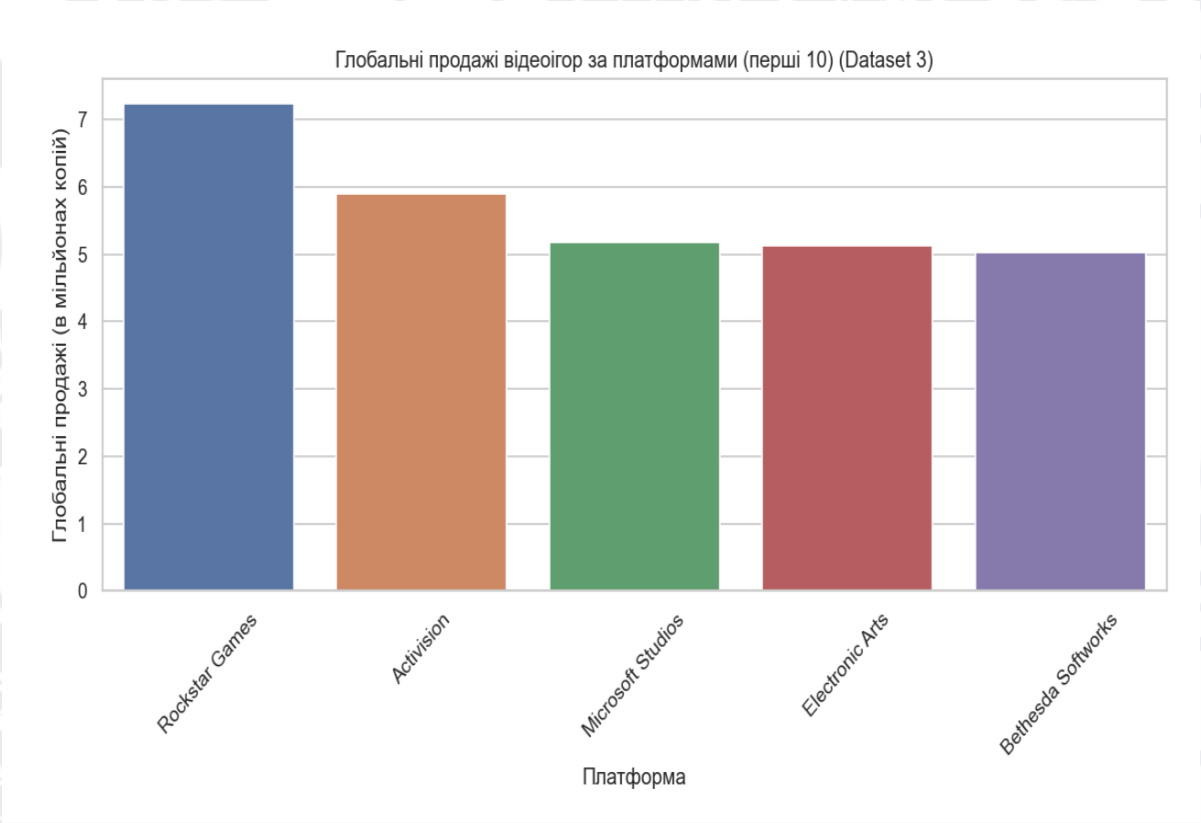

Рисунок 3.8 – Глобальні продажі відеоігор за платформами

Для того, щоб аналізувати та візуалізувати загальні продажі відеоігор за роками у різних регіонах створюється список regions, який містить назви регіонів, такі як: "NA_Sales" (продажі в Північній Америці), "JP_Sales" (продажі в Японії), "EU_Sales" (продажі в Європі) і "Other_Sales" (продажі в інших регіонах). Групуються дані за роками (Year_of_Release) та підсумовуються продажі у кожному з регіонів (стовпці regions). reset_index() використовується для повернення групованих даних до формату DataFrame.

Створено графік розподілу продажів за роками для різних регіонів за допомогою бібліотеки Plotly Express (go.Figure() та go.Scatter()). Для кожного регіону створюється лінійний графік, де по осі «x» - роки, а по осі «y» - сума продажів у відповідному регіоні.

```
geo_tdf = pd.read_csv("C:\\Users\\Vlad\\OneDrive\\Робочий стіл\\Video_sales\\Video_Games_Sales_as_at_22_Dec_2016.csv")
regions = ['NA_Sales', 'JP_Sales', 'EU_Sales', 'Other_Sales']
region_sales_sufix = '_Sales'
geo_tdf.rename(columns={region + region_sales_sufix: region for region in regions}, inplace=True)
grouped_data = geo_tdf.groupby(['Year_of_Release'])[regions].sum().reset_index()
fig = go.Figure()
for region in regions:
    fig.add_trace(go.Scatter(
        x=grouped_data['Year_of_Release'],
        y=grouped_data[region],
        mode='lines',
        name=region,
    ))
fig.update_layout(title="Загальні продажі за роками по регіонах (мільйони копій)")
fig.update_xaxes(type='category')
fig.show()
```

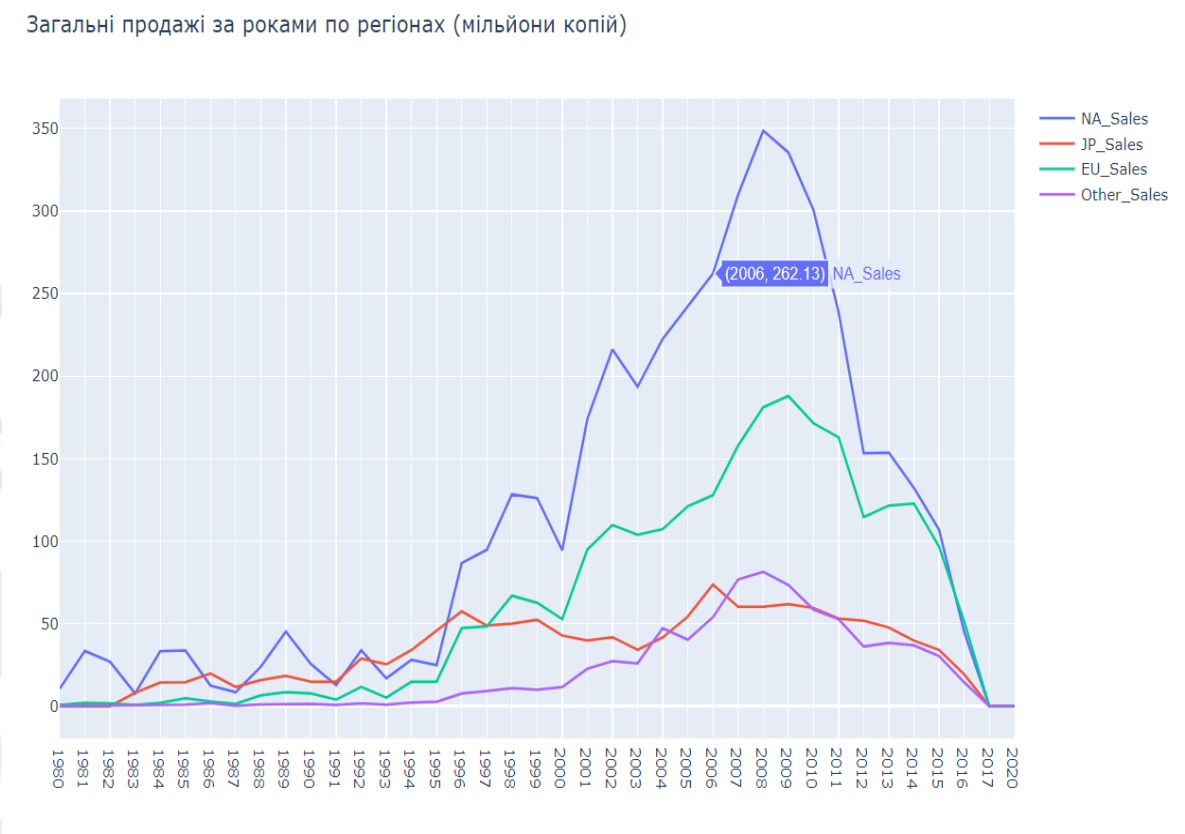

Рисунок 3.9 – Загальні продажі за роками по регіонах

Для виведення даного та наступних графіків використовується браузер. Також можна побачити, що при наведені на якусь із точок на графіку він показує нам рік та кількість проданих копій у той рік який було обрано. Ще можливо відключити будь який графік, щоб наприклад більш детально роздивитись графік продажу у одному регіоні, що буде більш наглядно.

Далі було виконано аналіз та візуалізацію розподілу продажів відеоігор за роками у різних регіонах. Для кожного року обчислюються суми продажів в кожному регіоні, і ці значення додаються як нові стовпці: NA_mean (сума продажів в Північній Америці), EU_mean (сума продажів в Європі), JP_mean (сума продажів в Японії), Other_mean (сума продажів в інших регіонах).

Далі дані готуються для створення анімованої гістограми розподілу продажів за регіонами для кожного року:

- створюються чотири додаткові DataFrames: temp_df1, temp_df2, temp_df3, temp_df4, кожен з яких містить інформацію про

розподіл продажів для певного регіону (Північна Америка, Європа, Японія, інші регіони) та певного року.

- за допомогою pd.concat об'єднуються чотири DataFrames в один final, який містить дані для всіх регіонів та всіх років.

Створюється анімована гістограма (fig=px.bar(...)) за допомогою бібліотеки Plotly Express. Графік відображає зміни в розподілі продажів за роками для кожного регіону. Анімація дозволяє спостерігати, як змінювалися продажі відеоігор за роками у різних регіонах. Код:

```
year_geo_df = dataset1 [["Year_of_Release",'NA_Sales','EU_Sales','JP_Sales','Other_Sales']]
year_geo_df[['NA_mean','EU_mean','JP_mean','Other_mean']] = year_geo_df.groupby('Year_of_Release')[['NA_Sales','EU_Sales','JP_Sales','Other_Sales']].transform('sum')
year_geo_df = year_geo_df.drop(['NA_Sales','EU_Sales','JP_Sales','Other_Sales'], axis=1)
year_geo_df = year_geo_df.drop_duplicates()
year_geo_df = year_geo_df.sort_values("Year_of_Release")
temp_df1 = pd.DataFrame({'Place': ['NA_Sales']*year_geo_df.shape[0], 'Year':year_geo_df['Year_of_Release'], 'Sales': year_geo_df['NA_mean']})
temp_df2 = pd.DataFrame({'Place': ['EU_Sales']*year_geo_df.shape[0], 'Year': year_geo_df['Year_of_Release'], 'Sales': year_geo_df['EU_mean']})
temp_df3 = pd.DataFrame({'Place': ['JP_Sales']*year_geo_df.shape[0], 'Year': year_geo_df['Year_of_Release'], 'Sales': year_geo_df['JP_mean']})
temp_df4 = pd.DataFrame({'Place': ['Other_Sales']*year_geo_df.shape[0], 'Year': year_geo_df['Year_of_Release'], 'Sales': year_geo_df['Other_mean']})
final = pd.concat([temp_df1,temp_df2,temp_df3,temp_df4], axis=0)
final = final.sort_values("Year")
fig=px.bar(
  final,
  x='Place',
  y="Sales",
  animation_frame="Year",
  animation_group="Place",
  color="Place",
  hover_name="Place",
  range_y=[0, 200]
)
fig.update_layout(title="Year sales distribution by region",title_x=0.5)
```

```
fig.show()
```

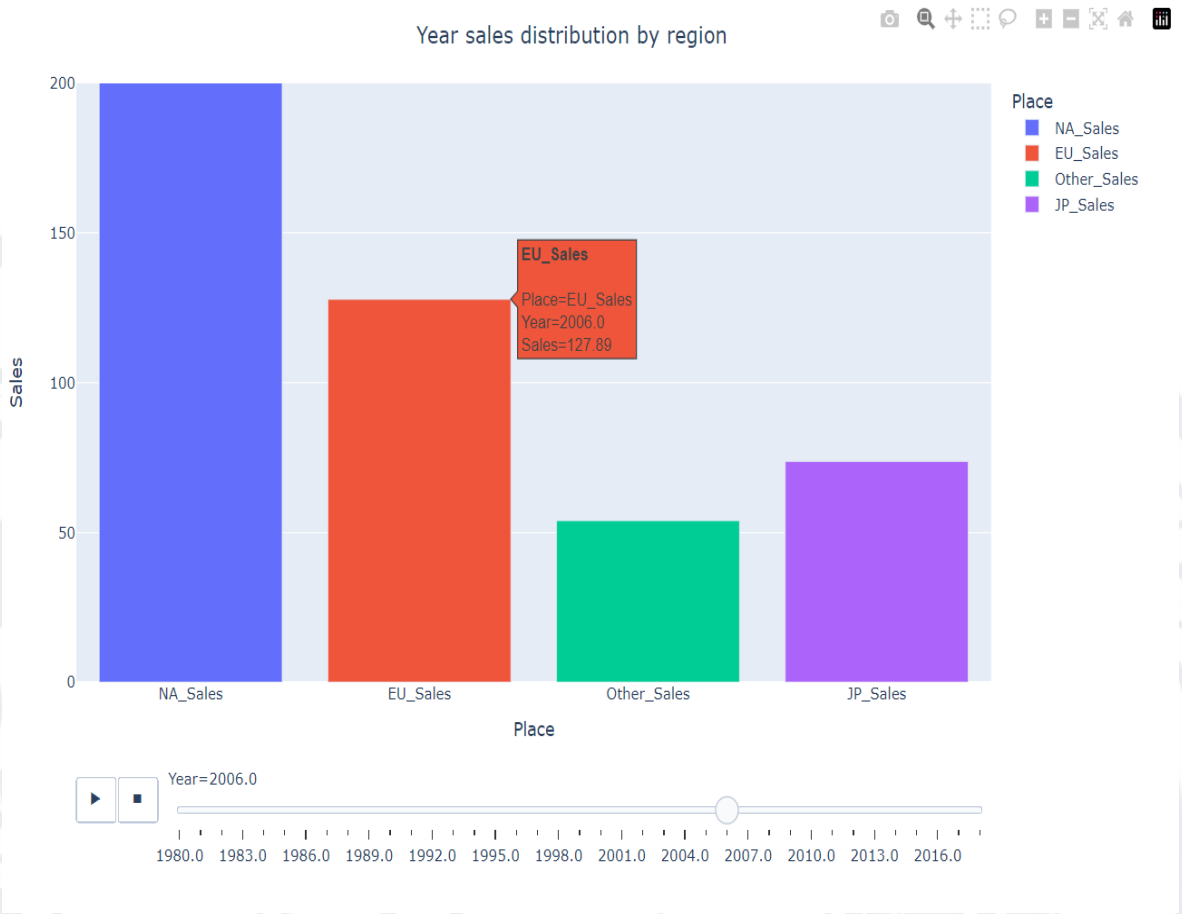

Рисунок 3.10 – Продажі за роками у різних регіонах

Графік має анімований ефект, що дозволяє спостерігати зміни у розподілі продажів відеоігор з плином часу. Кожен кадр анімації представляє один рік, і графік автоматично переходить від одного року до наступного, відображаючи, як розподіл продажів змінюється з року в рік. Графік є інтерактивним, тобто ви можете навести курсор миші на кожний стовпець, і отримати більше інформації про регіон та кількість продажів для конкретного року.

Далі виконувалась візуалізація даних про продажі відеоігор за жанрами. Було використано бібліотеку Plotly Express, щоб створити кругову діаграму (px.pie) на основі genre_tops_df. У цій діаграмі кожен сектор відповідає одному з жанрів, а розмір сектора відображає загальні продажі для цього жанру. Також було додано різні оформлення до діаграми, такі як: положення тексту

всередині секторів, відображення відсотків, кольори секторів і так далі. Отже, даний фрагмент відображає, які жанри відеоігор мають найбільший вплив на загальні продажі та демонструє цю інформацію у вигляді кругової діаграми з відсотковими відображеннями. Код продемонстовано далі:

```
df_genre_tots = dataset1[['Genre', 'Global_Sales']]
genre_tops = ['Action', 'Sports', 'Shooter', 'Role-Playing', 'Platform', 'Misc', 'Racing', 'Fighting', 'Simulation']
genre_tops_df = df_genre_tots[df_genre_tots['Genre'].isin(genre_tops)]
fig = px.pie(genre_tops_df,
             values='Global_Sales',
             names='Genre',
             title='Population of European continent',
             hover_data=['Genre'],
             labels={'lifeExp':'Video Games Genres'},
             hole=0.3,
            )
fig.update_traces(textposition='inside', textinfo='percent+label')
fig.show()
fig.update_layout(legend=dict(title='Genres'))
fig.update_traces(marker=dict(colors=['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd', '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22']))
fig.update_layout(autosize=False, margin=dict(l=0, r=0, b=0, t=0))
fig.update_traces(textinfo='label+percent', textfont_size=12)
fig.update_layout(
  title_text='Продажі відеоігор за жанрами',
  title_x=0.5,  # Вирівнюємо заголовок по центру
  title_font_size=20,
  title_font_family='Arial',
  annotations=[
    dict(
      text='Дані базуються на загальних продажах у мільйонах копій',
      showarrow=False,
      xref='paper',
      yref='paper',
      x=0.5,
      y=-0.1
    )
```

]
)

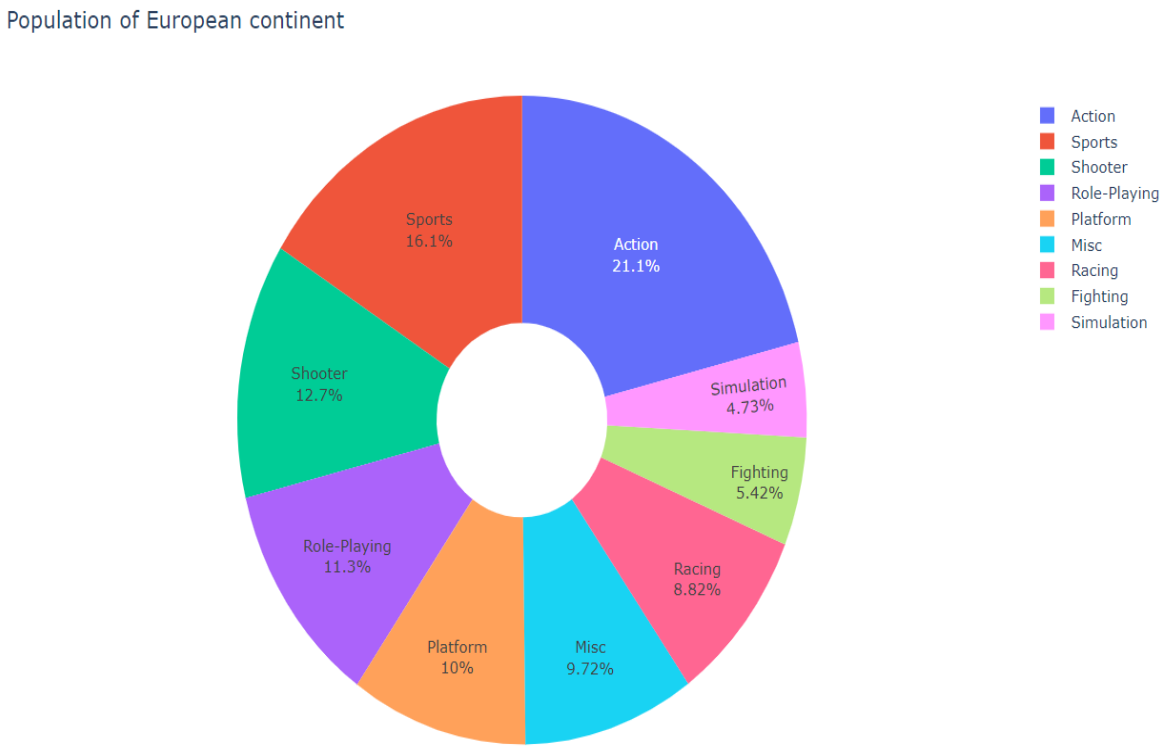

Рисунок 3.11 – Продажі відеоігор за жанрами

Головна ідея графіка полягає в тому, щоб візуально показати, які жанри відеоігор є найбільш популярними серед гравців і як вони розподілені за обсягом продажів. Такий підхід дозволяє швидко і зрозуміло оцінити, які жанри є найбільшими "гравцями" на ринку відеоігор. Також на даному графіку можливе відключення будь якого графіка щоб проаналізувати продажі без якогось одного або кілька жанрів.

3.2 Інтелектуальний аналіз продажу відеоігор

Далі у роботі використано рекурентну нейронну мережу Lstm (Long Short - term Memory) - це тип архітектури нейронної мережі, спеціально розроблений для ефективного аналізу та моделювання послідовностей даних у часових рядах.

LSTM використовує стан комірки для зберігання та передачі інформації протягом тривалих інтервалів часу. Це дозволяє моделі досліджувати та зберігати довгострокові залежності від даних. LSTM має Forget Gate, Update Gate, та Output Gate. Ці Gate регулюють потік інформації та дають змогу LSTM вирішувати, яку інформацію варто зберігати, забувати чи виводити. LSTM використовує ітерації для обробки послідовностей даних. Відомо, що повторення дозволяє моделі обробляти дані в контексті послідовності. LSTM розрізняє довготривалу та короткочасну пам'ять. Для того, щоб використовувати LSTM при роботі з часовими рядами, вхідні дані зазвичай подаються як послідовність кроків часу, кожен з яких представляє значення в певний момент часу. За допомогою LSTM модель може ефективно вивчати закономірності цих послідовностей та узагальнювати їх для прогнозування майбутніх значень.

Метод рекурентної нейронної мережі Lstm був обраний за його ефективність при роботі з часовими рядами і за можливості, які він надає для аналізу довгострокових і не стаціонарних залежностей даних. LSTM-це повторювана нейронна мережа, яка може ефективно моделювати та враховувати довгострокову залежність часових рядів. Це особливо важливо для завдань прогнозування, де важливі тенденції та закономірності можуть мати довгострокову динаміку. LSTM використовує внутрішню пам'ять для зберігання та оновлення інформації та може взаємодіяти з попередніми даними з часом. Це важливо для ефективного використання інформації за минулі періоди для прогнозування майбутніх значень. LSTM добре адаптований до аналізу нестаціонарних часових рядів, але його характеристики можуть змінюватися з часом. Це дозволяє моделі ефективно адаптуватися до змін у динаміці продажів відеоігор. LSTM може ефективно обробляти послідовності різної довжини, що дає можливість гнучко використовувати його для прогнозування завдань. Це буде важливим фактором, враховуючи, що дані про продажі можуть мати змінну тривалість часових інтервалів.

Існують інші методи прогнозування, такі як Arima та експоненціальне згладжування, але вибір методу залежить від деталей завдання та властивостей даних. У цьому випадку, враховуючи характер часових рядів продажів відеоігор, LSTM видається розумним вибором для отримання точних та гнучких прогнозів.

Основними етапами цього методу є:

- використання історичних даних про продажі відеоігор, щоб створити набір даних, що містить взаємозв'язок між часом та глобальними продажами;
- масштабування даних, важливий крок, що дозволяє нормалізувати значення продажів і присвоїти їм діапазон від 0 до 1. Це полегшує навчання моделі та підвищує стабільність;
- виконане розділення масштабованих даних на навчальні та тестові набори для ефективної оцінки моделі;
- використано функцію Create_dataset для створення вхідних та вихідних даних для навчання та тестування на основі кількості кроків часу;
- використано бібліотеку Keras для створення навчання моделі LSTM з одним шаром LSTM та одним шаром Danse;
- використано навчену модель для прогнозування продажів на тренувальному та тестовому наборах;
- відновлено масштабування отриманих прогнозів та побудовано графік з прогнозом майбутніх продаж.

Отож було створено модель LSTM та зроблено тренування моделі.

```
model = Sequential()
model.add(LSTM(units=50, input_shape=(1, time_steps)))
model.add(Dense(units=1))
model.compile(optimizer='adam', loss='mean_squared_error')
# Тренування моделі
```

model.fit(trainX, trainY, epochs=50, batch_size=1, verbose=2)

Ми маємо epochs, epochs глибокого навчання показує, що модель нейронної мережі проходить через весь навчальний набір даних за 1 повний шлях. Коли викликається метод fit для навчання моделі, весь навчальний набір даних подається в модель певну кількість разів (епоха). Кожен шлях через навчальний набір складається з декількох ітерацій, в ході яких модель коригує себе, щоб поліпшити узагальнене навчання.

Epoch – важливе поняття в навчанні нейронних мереж, вибір яких впливає на ефективність і здатність до навчання моделі. Якщо кількість Epoch занадто мала, модель, можливо, не зможе оптимізувати ваги для унікальних характеристик навчального набору даних. У моделі може не вистачити часу на "вивчення" слабких закономірностей в даних, тому її прогностична здатність може залишатися низькою. Параметр Batch_size визначає кількість вибіркових даних, що використовуються для оновлення однієї ваги. Більше значення може прискорити навчання, але для цього може знадобитися більше пам'яті. Менше значення може призвести до більш точних оновлень, але це може призвести до більш тривалих періодів навчання.

```
27/27 - 3s - loss: nan - 3s/epoch - 118ms/step
Epoch 2/50
27/27 - 0s - loss: nan - 98ms/epoch - 4ms/step
Epoch 3/50
27/27 - 0s - loss: nan - 85ms/epoch - 3ms/step
Epoch 4/50
27/27 - 0s - loss: nan - 109ms/epoch - 4ms/step
Epoch 5/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 6/50
27/27 - 0s - loss: nan - 86ms/epoch - 3ms/step
Epoch 7/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 8/50
27/27 - 0s - loss: nan - 109ms/epoch - 4ms/step
Epoch 9/50
27/27 - 0s - loss: nan - 108ms/epoch - 4ms/step
Epoch 10/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 11/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 12/50
27/27 - 0s - loss: nan - 93ms/epoch - 3ms/step
Epoch 13/50
27/27 - 0s - loss: nan - 76ms/epoch - 3ms/step
Epoch 14/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 15/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 16/50
27/27 - 0s - loss: nan - 110ms/epoch - 4ms/step
Epoch 17/50
27/27 - 0s - loss: nan - 125ms/epoch - 5ms/step
Epoch 18/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 19/50
27/27 - 0s - loss: nan - 94ms/epoch - 3ms/step
Epoch 20/50
```

Рисунок 3.12 – Тренування моделі (Epoch)

Після даних дій було побудовано графік з прогнозованими продажами за допомогою моделі LSTM.

```
all_dates = np.concatenate([extended_data.index.to_numpy(), future_dates.to_numpy()])
all_values = np.concatenate([scaler.inverse_transform(scaled_extended_data).flatten(), futurePredictPlot])
plt.plot(all_dates, all_values, label='Прогнозовані продажі', linestyle='--', color='red')
plt.title('Глобальні продажі відеоігор за роками')
plt.xlabel('Рік')
plt.ylabel('Глобальні продажі (в мільйонах)')
plt.legend()
plt.show()
```

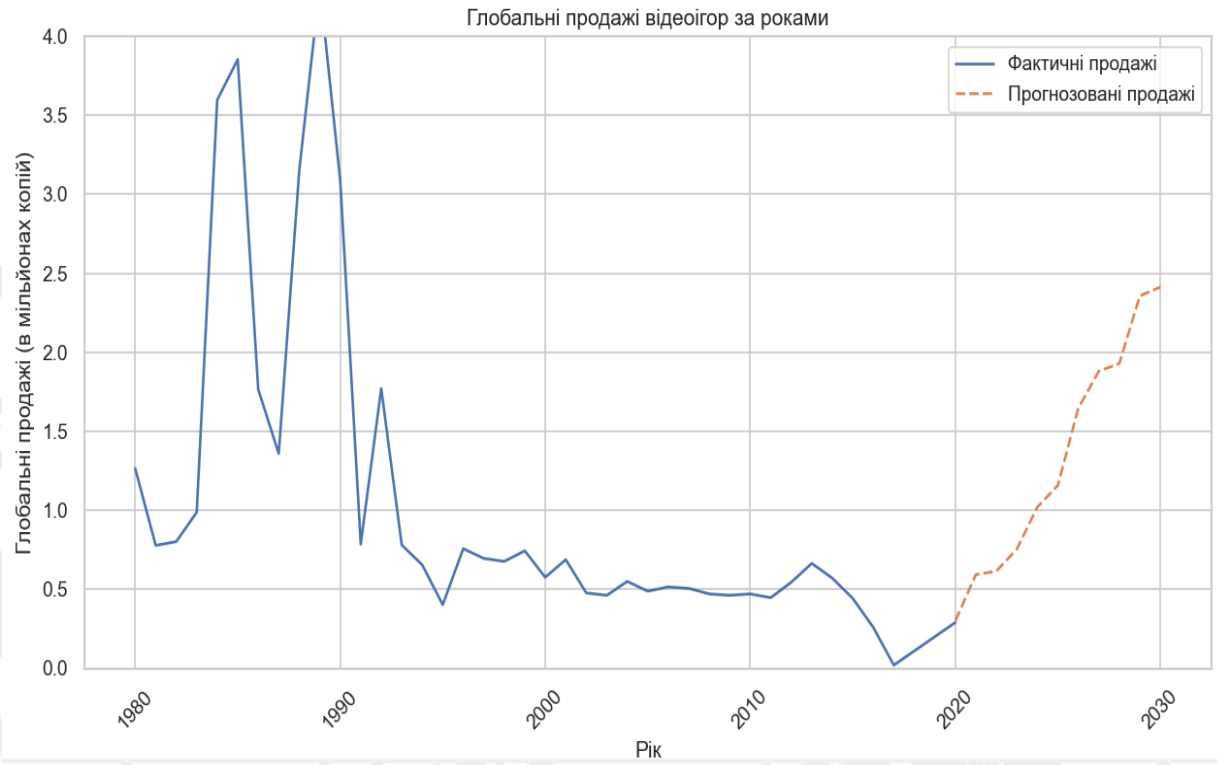

Рисунок 3.13 – Прогноз продаж за допомогою нейронної мережі LSTM

Прогнозування глобальних продажів відеоігор за допомогою повторюваної нейронної мережі LSTM призначене для кількох цілей і може призвести до важливих висновків. Прогноз дозволяє компаніям індустрії відеоігор розробляти стратегії на майбутнє. Знання очікуваного обороту може вплинути на виробничі, маркетингові та рекламні рішення. Результати прогнозування можуть допомогти виявити тенденції переваг споживачів у індустрії відеоігор. Наприклад, якщо прогноз передбачає збільшення продажів певного типу ігор, це може свідчити про популярність певного жанру чи концепції. Розуміння того, як зміниться продаж у майбутньому, може допомогти розробити та налаштувати маркетингові стратегії. Компанії можуть адаптувати свої кампанії відповідно до очікуваних тенденцій. Прогноз може визначити оптимальний період для запуску нової гри або важливої події в індустрії відеоігор. Наприклад, якщо очікується висока популярність протягом певного періоду часу, це може бути важливим для планування випуску нового продукту. Якщо прогноз передбачає негативну тенденцію, компанії можуть

заздалегідь підготуватися до можливих викликів, вживаючи відповідних заходів, таких як скорочення виробництва або зміна маркетингових стратегій.

Загальна мета – забезпечити більшу впевненість у майбутніх рішеннях та вирішити проблеми планування та управління відеоіграми. Цей прогноз дозволяє компаніям більш адаптивно та ефективно реагувати на мінливий ринок.

3.3 Використані методи та аналіз датасета

В роботі використовуються різні методи інтелектуального аналізу даних для розуміння та візуалізації інформації з трьох датасетів, що містять інформацію про продажі відеоігор. Використані методи перераховані нижче.

Один із методів це є огляд та обробка даних:

- pd.read_csv("file_path"): Завантаження даних з CSV-файлів відбувається за допомогою бібліотеки Pandas.;
- dataset1.head(), dataset2.head(), dataset3.head(): Виведення перших рядків для перевірки структури даних;
- dataset1.describe(), dataset2.describe(), dataset3.describe(): Виведення описової статистики для кожного набору даних.

Наступним методом є обчислення сумарних продажів:

- total_sales1 = dataset1['Global_Sales'].sum(): Обчислення загальних обсягів продажів для першого набору даних.

Вибір найкращої гри за продажами:

- best_selling_game1 = dataset1.loc[dataset1['Global_Sales'].idxmax()]: Вибір найкращої гри за продажами для першого набору даних.

Наступним методом була візуалізація даних за допомогою Seaborn та Matplotlib, використання виконувалось наступним чином:

- використання sns.lineplot для візуалізації графіка продажів відеоігор за роками для першого набору даних;

- використання sns.barplot для візуалізації графіка продажів за жанрами для другого набору даних;
- використання sns.barplot для візуалізації графіка продажів за платформами (перші 10) для третього набору даних.

Візуалізація даних за допомогою Plotly Express та Plotly Graph Objects:

- використання go.Figure() та go.Scatter для побудови графіка загальних продаж за роками по регіонах (Північна Америка, Японія, Європа, Інші);
- використання px.bar для анімації стовбчатої діаграми з продажами в різних регіонах за роками.

Візуалізація даних за допомогою Plotly Express (кругова діаграма):

- використання px.pie для побудови кругової діаграми продажів відеоігор за жанрами.

Також використовувались додаткові налаштування та стилізація графіків:

- sns.set(style="whitegrid"): Налаштування стилів для графіків Seaborn;
- налаштування заголовків, міток, розмірів графіків та інші елементи для забезпечення зручного відображення та читання інформації.

Розглянуті методи дозволяють здійснювати обробку, аналіз та візуалізацію даних, щоб краще зрозуміти їх розподіл та тенденції. Кожен метод виконує конкретні завдання для допомоги у розв’язанні загальної задачі аналізу даних.

Термін “інтелектуальний аналіз даних” охоплює широкий спектр методів та інструментів для розуміння, інтерпретації та використання даних для прийняття рішень. Розглянемо, які елементи роботи пов’язані з інтелектуальним аналізом даних. Для огляду та обробки даних інтелектуальний аспект включає такі особливості як: обробку даних, включаючи вивчення їх структури, визначення ключових параметрів та відсутніх значень. Для обчислення сумарних продажів інтелектуальним

аспектом можна назвати аналіз загальних продажів, який може допомогти визначити ключові тренди та успішність різних платформ та жанрів. Вибір найкращої гри за продажами визначає найуспішніші ігри та може вказувати на те, які фактори сприяли їхньому успіху, що може також вважатись як інтелектуальним аспектом. Візуалізація даних за допомогою Seaborn та Matplotlib містить наступний інтелектуальний аспект: візуалізація даних сприяє кращому розумінню зв'язків та патернів у даних. Візуалізація даних за допомогою Plotly Express та Plotly Graph Objects використовує інтерактивні графіки, що може полегшити розуміння та виявлення ключових трендів. Візуалізація даних за допомогою Plotly Express містить наступний інтелектуальний аспект: кругова діаграма може виділити частки ринку для різних жанрів в інтерактивній та легкозрозумілій формі.

Оцінка продажів відеоігор, проведена у роботі, включає в себе декілька аспектів, які допомагають зрозуміти ринкові та індустріальні тенденції в галузі відеоігор.

Загальні продажі та найкраща гра, яку можна побачити на рис. 3.5 здійснює обчислення загальних продажів для першого набору даних, що дозволяє отримати уявлення про сумарний обсяг продажів відеоігор а також найкращу гру за продажами, також наведено всю інформацію про дану гру, що дозволяє більш детально розглянути всі аспекти. Отже загальні продажі дозволяють отримати уявлення про розмір ринку відеоігор та його динаміку.

Аналіз трендів за роками який можна побачити на рис. 3.6 використовує графіки для візуалізації змін продажів відеоігор за роками, що дозволяє виявити тренди та патерни розвитку галузі. Ці графіки допомагають виявити зміни та тренди у відеоігровій індустрії з плином часу, що може бути корисно для стратегічного планування. Так, на рис. 3.4, коли приблизно з 1983 по 1990 рік було найбільше продажів та найбільший прорив у відео продажах, після чого продажі різко впали та коливались у подальших роках більш стабільно, але такого прориву як у ті роки більше не спостерігалось.

Аналіз за жанрами та платформами, який наведено на рис. 3.7 та рис. 3.6, використано стовбчасті діаграми для отримання інформації про популярність видів ігор та платформ. Цей аналіз вказує на популярні жанри та платформи, що може бути використано для орієнтації в розробці нових продуктів чи визначенні маркетингових стратегій. Так, аналіз рис. 3.8 вказує, що за жанрами більш популярні є такі жанри як: спорт, шутери та пригодницькі ігри, тому можна з певністю сказати, що ігри в даній тематиці будуть більше продаватись. Тоді, як з платформами не так все однозначно, тому що вони йдуть майже на одному рівні, тільки Rockstar Games більше відірвався від конкурентів.

Регіональний аналіз зображено на рис. 3.9, у ньому проведено аналіз продажів в різних регіонах (Північна Америка, Японія, Європа, інші) з використанням інтерактивних графіків. Це надає інформацію про різниці в споживчих звичках у різних регіонах, що корисно при адаптації продуктів та маркетингових кампаній. З графіка можна побачити, що найбільше продажів припадає на Америку та Європу, тому виробник повинен бути більш сконцентрований на даних регіонах, щоб отримати найбільше продажів. Також на графіку є кількість проданих копій у конкретний рік та за конкретним регіоном. На рис. 3.10 можна побачити графік, який показує продажі у різних регіонах за роками. Переглянувши даний графік можна спрогнозувати, який регіон був і буде перспективним у подальшому, а на якому можна менш сконцентровуватись.

Аналіз найкращих жанрів, який продемонстровано на рис. 3.11 використано кругову діаграму для виділення часток ринку найпопулярніших жанрів в інтерактивній формі. Цей графік може допомогти визначити, які жанри є найпопулярнішими серед споживачів, допомагаючи спрямувати зусилля в розвитку конкретних типів ігор. Тут також можна побачити, що найбільш популярними жанрами є шурети, спорт та пригодницькі, тому варто звернути увагу на ці жанри та розвивати ігри у даному напрямку.

На рис. 3.13 зображено прогноз продаж до 2030 року, який здійснено за допомогою рекурентної нейронної мережі LSTM. Цей графік показує, що прогнозується збільшення продаж, тому варто прийняти це до уваги та збільшити виготовлення ігор.

Розглянуте допомагає детально вивчити різні аспекти ринку відеоігор та визначити основні фактори, що впливають на продажі. Це допомагає у формулюванні стратегій відеоігор, плануванні маркетингових кампаній та прийнятті стратегічних рішень в індустрії розробки відеоігор. Наприклад, знання про популярні жанри та платформи може допомогти розробникам створювати продукти, які відповідають попиту ринку, а регіональний аналіз може підтримати глобальні маркетингові стратегії.

Висновки до розділу

У даному розділі було проведено аналіз продажів відеоігор. Було завантажено дані із трьох датасетів про продажі відеоігор. Графічно зображено даний аналіз та зроблено прогноз продажів. Вказано як саме впливає прогноз та як варто використовувати дані взяті із даного аналізу.

ВИСНОВКИ

В ході виконання магістерської роботи було докладно вивчено мову програмування Python, та її бібліотеки: pandas, matplotlib.pyplot, seaborn, plotly.express та plotly.graph_objects.

Було розглянуто актуальність теми та постановку задачі. Також було обрано два аналога аналізу та візуалізації продажів відеоігор, розглянуто їх переваги та недоліки.

Основним етапом був аналіз та візуалізації продажів відеоігор. Розглянуто способи та графіки для аналізу та візуалізації.

Було виконано інтелектуальний аналіз даних, під час якого модель навчається на даних датасетів, після навчання модель прогнозує можливий напрямок продажів відеоігор, що дало змогу побачити чи варто розробляти ігри у більш стрімкому русі. Представлені результати підкреслюють практичне застосування отриманої інформації для індустрії відеоігор та підтверджують корисність вивчених методів та інструментів аналізу та візуалізації.

Обрані методи дослідження, такі як: аналіз та візуалізації, дозволили ефективно вирішити поставлені завдання з вивчення продажів відеоігор.

Подальші удосконалення програми можуть включати в себе розширення функціональності з урахуванням додаткових параметрів, таких як: оцінки критиків, популярність в інших регіонах. Розробка інтерактивного інтерфейсу для кінцевого користувача може сприяти простоті використання програми.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Актуальність аналізу продажу відеоігор URL: https://investory.news/rinok-kompyuternix-igor-zris-do-majzhe-150-milyardiv/ (дата звернення: 04.05.2023).

2. SteamSpy URL: https://steamspy.com/ (дата звернення: 04.05.2023).

3. Python URL: https://acode.com.ua/intro-python/ (дата звернення: 10.05.2023).

4. Історія розвитку Python URL: https://vgoru.org/pererva-na-kavu/python-mova-programuvannya-majbutnogo (дата звернення: 10.05.2023).

5. Вплив інших мов програмування на Python URL: https://vuzlit.com/1018420/vpliv_inshih_python (дата звернення: 10.05.2023).

6. Переваги та недоліки Python URL: https://futurenow.com.ua/perevagy-ta-nedoliky-python/ (дата звернення: 10.05.2023).

7. pandas URL: https://cloud.itstep.org/blog_3/obvious-errors-in-python-pandas-when-working-with-big-data (дата звернення: 17.05.2023).

8. Matplotlib URL: https://jait.donnu.edu.ua/article/view/12280 (дата звернення: 17.05.2023).

9. Функції Matplotlib URL: https://matplotlib.org/stable/users/index.html (дата звернення: 17.05.2023).

10. Linux-console URL: https://uk.linux-console.net/?p=8214#gsc.tab=0 (дата звернення: 23.05.2023).

11. Seaborn URL: https://seaborn.pydata.org/ (дата звернення: 23.05.2023).

12. NumPy URL: https://devzone.org.ua/post/comu-vam-slid-vikoristovuvati-numpy (дата звернення: 08.09.2023).

13. Keras URL: http://cloud-5.bitp.kiev.ua/?page_id=600 (дата звернення: 08.09.2023).

14. Keras URL: http://cloud-5.bitp.kiev.ua/?page_id=600 (дата звернення: 08.09.2023).

15. API глибокого навчання Keras 3.0 URL: https://proit.org.ua/api-ghlibokogho-navchannia-keras-3-0-pidtrimuie-tensorflow-pytorch-jax/ (дата звернення: 08.09.2023).

16. Офіційна документація TensorFlow/Keras URL: https://www.tensorflow.org/api_docs/python/tf/keras (дата звернення: 04.05.2023).

17. Офіційна документація Python URL: https://docs.python.org/3/ (дата звернення: 16.09.2023).

18. Використання Python URL: https://foxminded.ua/de-vykorystovuietsia-python/ (дата звернення: 16.09.2023).

19. Особливості Python URL: https://spacelab.ua/articles/osobennosti-i-preimushestva-python/ (дата звернення: 16.09.2023).

20. Real Python Tutorials URL: https://realpython.com/ (дата звернення: 16.09.2023).

21. Специфікації Python URL: https://www.geeksforgeeks.org/python-programming-language/ (дата звернення: 16.09.2023).

22. Бібліотеки Python URL: https://awesome-python.com/ (дата звернення: 18.09.2023).

23. Візуалізація даних на Python URL: https://towardsdatascience.com/introduction-to-data-visualization-in-python-89a54c97fbed (дата звернення: 18.09.2023).

24. Загальна документація та приклади Pandas URL: https://pandas.pydata.org/pandas-docs/stable/ (дата звернення: 27.09.2023).

25. Використання Pandas URL: https://pandas.pydata.org/pandas-docs/stable/user_guide/cookbook.html (дата звернення: 27.09.2023).

26. Документація та використання Matplotlib URL: https://matplotlib.org/stable/users/index.html (дата звернення: 06.10.2023).

27. Візуалізація з Matplotlib URL: https://towardsdatascience.com/matplotlib-tutorial-learn-basics-of-pythons-powerful-plotting-library-b5d1b8f67596 (дата звернення: 06.10.2023).

28. Документація Seaborn URL: https://seaborn.pydata.org/ (дата звернення: 09.10.2023).

29. Налаштування графіків у Seaborn URL: https://towardsdatascience.com/data-visualization-using-seaborn-fc24db95a850 (дата звернення: 09.10.2023).

30. Продажі відеоігор URL: https://www.statista.com/topics/868/video-games/#topicOverview (дата звернення: 15.10.2023).

31. Аналіз продажів відеоігор URL: https://newzoo.com/ (дата звернення: 15.10.2023).

32. Документація NumPy URL: https://numpy.org/doc/stable/ (дата звернення: 18.10.2023).

33. Основи NumPy URL: https://www.w3schools.com/python/numpy/numpy_intro.asp (дата звернення: 18.10.2023).

34. Використання NumPy URL: https://github.com/rougier/numpy-100 (дата звернення: 18.10.2023).

35. PEP in Python: https://peps.python.org/pep-0001/ (дата звернення: 20.10.2023).

36. Документація Plotly URL: https://plotly.com/python/ (дата звернення: 20.10.2023).

37. Можливості Plotly URL: https://towardsdatascience.com/plotly-python-open-source-graphing-library-ecea3ef5c5f9 (дата звернення: 20.10.2023).

38. використання Plotly Express URL: https://plotly.com/python/plotly-express/ (дата звернення: 20.10.2023).

39. Приклади робіт Plotly URL: https://github.com/plotly/plotly.py (дата звернення: 20.10.2023).

40. Гороховатський В. О., Творошенко І. С. Методи інтелектуального аналізу та оброблення даних: навч.посібник – Харків: ХНУРЕ, 2021. – 92с.

41. Візуалізація за допомогою Matplotlib та Seaborn URL: https://jakevdp.github.io/PythonDataScienceHandbook/04.00-introduction-to-matplotlib.html (дата звернення: 24.10.2023).

42. документація бібліотеки TensorFlow URL: https://www.tensorflow.org/ (дата звернення: 24.10.2023).

43. Аналіз та прогнозування у Python URL: https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b (дата звернення: 24.10.2023).