Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Факультет інформаційних технологій
(факультет)

Кафедра системного аналізу та управління
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня бакалавра

Здобувача вищої освіти Амбарцумяна Сергія Павловича

академічної групи 124-21-2

спеціальності 124 Системний аналіз

за освітньо-професійною програмою Системний аналіз

на тему: «Аналіз і обробка ключових показників аккаунту гравця у комп'ютерній грі для пошуку аномальних даних»

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*ас. Шевченко Ю.О*			
розділів:				
Інформаційно–аналітичний	*ас. Шевченко Ю.О*			
Спеціальний розділ	*ас. Шевченко Ю.О.*			
Рецензент	*д.т.н., проф. Новицький І.В.*			
Нормоконтролер	*к.ф.–м.н., доц. Хом'як Т.В.*			

Дніпро
2025

ЗАТВЕРДЖЕНО:
завідувач кафедри
Системного аналізу та управління
(повна назва)

_____ *к.т.н., доц. Желдак Т.А.*
(підпис) (прізвище,ініціали)

«____» _____20___року

ЗАВДАННЯ
на кваліфікаційну роботу
ступеня *бакалавра*

здобувачу вищої освіти *Амбарцумяну С. П.* **академічної групи** *124–21–2*
спеціальності: *124 Системний аналіз*
за освітньо-професійною програмою: *Системний аналіз*
на тему «*Аналіз і обробка ключових показників аккаунту гравця у комп'ютерній грі для пошуку аномальних даних»*
затверджену наказом ректора НТУ «Дніпровська політехніка» від *05.05.2025р. №336-с*

Розділ	Зміст	Терміни виконання
1. Інформаційно-аналітичний	*Проаналізувати структуру гри. Отримати дані про матчі гравця, описати і проаналізувати різні методи обробки даних та кластеризації.*	10.03.2025 – 01.05.2025
2.Спеціальний	*Вибрати більш ефективний метод кластеризації, написати програму, яка реалізує цей метод для пошуку аномалій в грі Dota 2.*	01.04.2025-10.06.2025

Завдання видано _____ *Шевченко Ю.О.*
(підпис) (прізвище, ініціали)

Дата видачі: *06.03.2025*
Дата подання до екзаменаційної комісії: *24.06.2025*

Прийнято до виконання _____ *Амбарцумян С. П.*
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 45 с., 22 рис., 2 табл., 5 додатків 13 джерел.

Об'єктом дослідження в роботі є гра «DOTA 2».

Предметом дослідження є статистика та історія матчів гравця.

Метою даної кваліфікаційної роботи є кластеризація матчів гравця для подальшого аналізу статистики та пошуку аномальних ігор.

Методи дослідження: методи кластеризації, алгоритми для пошуку аномалій, методи графічного відображення, csv та json для зберігання даних.

В інформаційно-аналітичному розділі було описано базові механіки гри, метод отримання даних, а також були проаналізовані різні методи кластеризації та підходи до аналізу ігрової активності, розглянуто їхні принципи роботи, результати застосування та основні недоліки.

У другому розділі детально описано фінальний обраний підхід до кластеризації, подано його результати та порівняння з попередніми методами, а також розроблено алгоритм для виявлення підозрілих матчів.

Практична цінність дослідження полягає у створенні інструменту, який дозволяє гравцям DOTA 2 проаналізувати свою ігрову статистику в розрізі часових періодів, визначити найуспішніші стратегії гри, а також виявити потенційні ситуації передачі акаунту іншим користувачам, що може вплинути на чесність змагань.

Ключові слова: КЛАСТИРІЗАЦІЯ, АНОМАЛІЇ, МАТЧ, СТАТИСТИКА, ГРАВЕЦЬ, DOTA, АНАЛІЗ.

ABSTRACT

Explanatory note: 45 p., 22 fig., 2 tables, 13 sources, 5 appendices.

The object of research is player's match history and gameplay statistics.

The subject of research is the company's internal business processes and Internet advertising mechanisms.

The purpose of this qualification work is to perform clustering of a player's matches for subsequent analysis of gameplay statistics and detection of anomalous games.

Research methods – clustering techniques, anomaly detection algorithms, graphical visualization methods, and structured data handling using CSV and JSON formats.

The *first chapter* describes the basic mechanics of the game, the data acquisition process, and analyzes various clustering methods and approaches to gameplay activity analysis. Their working principles, application results, and major limitations are discussed.

A *second chapter* provides a detailed explanation of the final selected clustering approach, its outcomes, comparison with previous methods, and introduces a developed algorithm for identifying suspicious matches.

The *practical value:* the developed tool enables Dota 2 players to analyze their gameplay statistics over time, identify their most effective strategies, and detect potential cases of account sharing, which may affect the fairness of competitive play.

Keywords: CLUSTERING, ANOMALIES, MATCHES, PLAYER STATISTICS, DOTA, ANALISYS.

ЗМІСТ

ВСТУП

У сучасному цифровому суспільстві комп’ютерні ігри відіграють значну роль не лише як засіб розваги, але й як складний інтерактивний простір для аналізу даних, поведінки користувачів і розробки інтелектуальних алгоритмів. Однією з найпопулярніших кіберспортивних дисциплін є гра Dota 2, яка завдяки великій кількості матчів і складній ігровій механіці генерує об’ємні масиви даних, що мають значний аналітичний потенціал.

Актуальність обраної теми зумовлена потребою у глибшому розумінні ігрової активності користувача, а також у можливості виявлення аномалій, пов’язаних із різкою зміною стилю гри, що може свідчити про передачу акаунта або використання стороннього програмного забезпечення. Аналіз історії матчів і статистики дозволяє виділити періоди з нетиповою поведінкою гравця, оцінити ефективність його стратегії в різні ігрові етапи, а також виявити потенційні порушення чесної гри.

Об’єкт дослідження – комп’ютерна гра Dota 2.

Предмет дослідження – історія матчів та статистика ігрових показників одного гравця.

Мета роботи – кластеризація матчів гравця з подальшим аналізом кластерів для виявлення аномальних ігор, що відхиляються від типових показників.

Методи дослідження: методи кластеризації, аналіз відхилень, алгоритми виявлення аномалій, методи візуалізації даних, опрацювання CSV та JSON-файлів.

У межах кваліфікаційної роботи було реалізовано:

- розробку алгоритму кластеризації ігрових сесій на основі розподілу героїв за періодами;
- виявлення кластерів із підвищеними показниками KDA (відношення вбивств, допомог та смертей);

- перевірку наявності подібних ігор у сусідніх кластерах з метою верифікації аномалій;
- побудову системи, яка дозволяє візуалізувати результати кластеризації та аномального аналізу.

Практичне значення отриманих результатів полягає у створенні інструменту, що дозволяє гравцям «Dota 2» проаналізувати свою активність у грі, виявляти нехарактерні періоди гри та підвищувати ефективність власних стратегій. Розроблений алгоритм також може бути використаний розробниками гри для створення спеціальних налаштувань для виявлення можливих порушень у грі з боку користувачів.

Апробація результатів кваліфікаційної роботи здійснена на II (VIII) Міжнародній науково-практичній конференції здобувачів вищої освіти і молодих учених «Інформаційні технології: теорія і практика» (2 – 4 квітня 2025 р., м. Запоріжжя – Харків – Дніпро, Україна). Отримані наукові результати відображено у публікації [2].

РОЗДІЛ 1 ІНФОРМАЦІЙНО-АНАЛІТИЧНИЙ

1.1 Базові механіки гри

Гра Dota 2 є однією з найпопулярніших багатокористувацьких онлайн-ігор жанру MOBA (Multiplayer Online Battle Arena), де дві команди по п’ять гравців змагаються одна з одною на мапі, поділеній на три основні лінії — верхню (Top), центральну (Mid) та нижню (Bottom).

Метою гри є знищення головної будівлі суперника — Ancient, що розташована в серці їхньої бази. Кожен гравець обирає унікального героя зі спеціальними вміннями (skills), характеристиками та роллю в команді:

- Carry — герой, що з часом стає найсильнішим.
- Support — підтримує команду: лікує, купує предмети, забезпечує бачення.
- Tank/Initiator — витримує шкоду, ініціює бої.
- Nuker, Disabler, Pusher та інші.

Основні статистичні показники для аналізу гри:

- Hero ID — унікальний числовий ідентифікатор обраного героя.
- Kills (Вбивства) — кількість вбитих ворожих героїв.
- Deaths (Смерті) — скільки разів герой гравця загинув.
- Assists (Асисти) — кількість допомог у вбивстві ворога.
- Date – дата коли матч був зіграний.

Основна з характеристик гри це KDA — коефіцієнт ефективності, який обчислюється за формулою:

$$KDA = \frac{Kills+Assists}{\max(1,Deaths)} \tag{1.1}$$

Усі ці фактори дозволяють аналізувати зміни в стилі гри користувача, виявляти періоди активного зростання або навпаки — зниження ефективності.

Ці показники є основою для подальшої обробки даних та кластеризації, яка дозволяє групувати матчі за схожими патернами використання героїв у часовому розрізі.

1.2 Отримання та зберігання інформації про матчі гравця

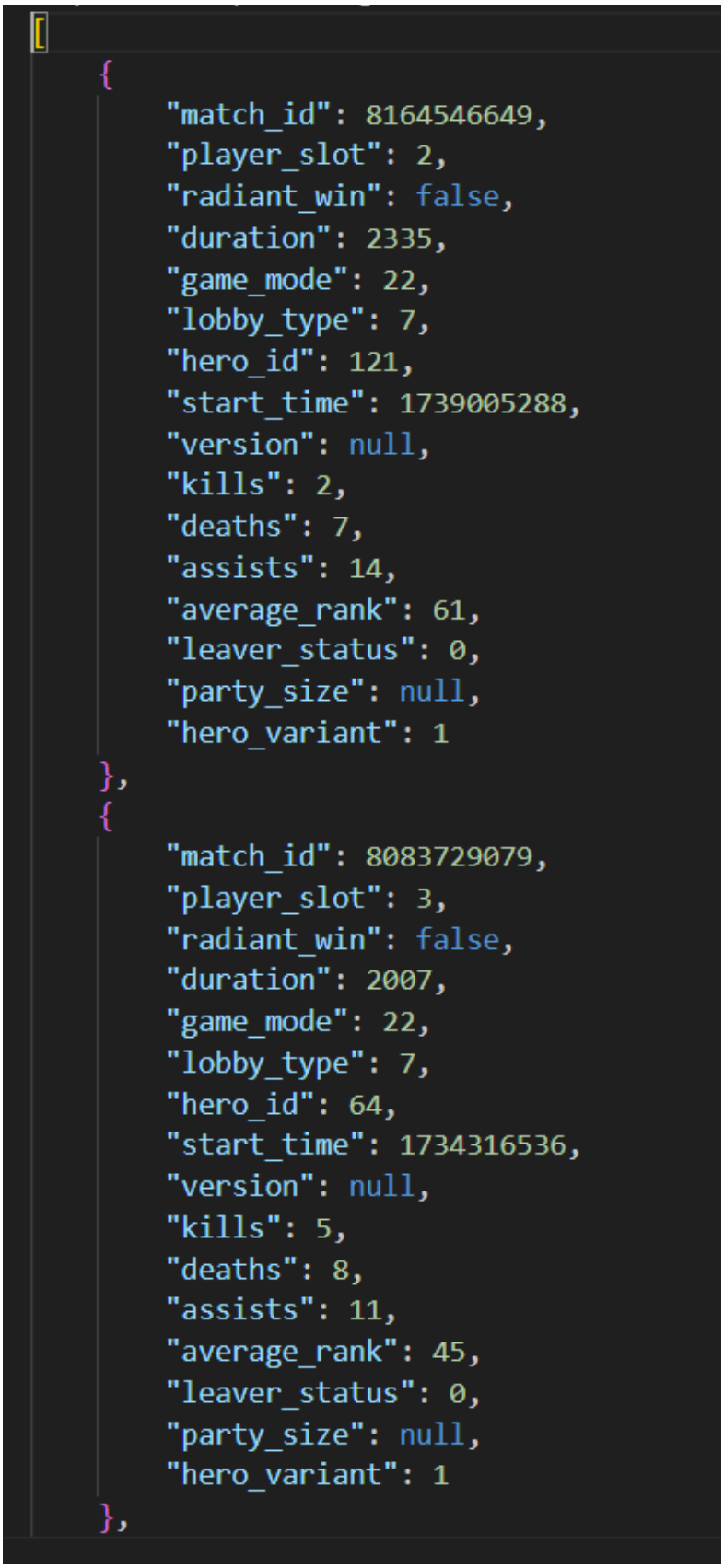
Для аналізу поведінки гравця в грі Dota 2 було необхідно отримати структуровані дані про його ігрову активність. З цією метою використано відкритий програмний інтерфейс *OpenDota API*, який надає доступ до історії матчів конкретного гравця за його *Steam ID*.

На першому етапі було реалізовано отримання інформації про історію матчів конкретного гравця за його Steam ID (рис. 1.1). Використано бібліотеку requests для надсилання HTTP-запиту до API та збереження результату у форматі JSON.

```python
import requests
import pandas as pd
import json


account_id = 447421252 #287156880 #359887171
url = f"https://api.opendota.com/api/players/{account_id}/matches"

response = requests.get(url)
print(response)
matches = response.json()

with open("dota_matches_sample.json", "w") as file:
    json.dump(matches, file, indent=4)

```

Рисунок 1.1 – Код для GET реквесту до OpenDota API

У результаті виконання коду формується локальний файл dota_matches_sample.json, який містить список матчів у структурованому вигляді.



```
[
    {
        "match_id": 8164546649,
        "player_slot": 2,
        "radiant_win": false,
        "duration": 2335,
        "game_mode": 22,
        "lobby_type": 7,
        "hero_id": 121,
        "start_time": 1739005288,
        "version": null,
        "kills": 2,
        "deaths": 7,
        "assists": 14,
        "average_rank": 61,
        "leaver_status": 0,
        "party_size": null,
        "hero_variant": 1
    },
    {
        "match_id": 8083729079,
        "player_slot": 3,
        "radiant_win": false,
        "duration": 2007,
        "game_mode": 22,
        "lobby_type": 7,
        "hero_id": 64,
        "start_time": 1734316536,
        "version": null,
        "kills": 5,
        "deaths": 8,
        "assists": 11,
        "average_rank": 45,
        "leaver_status": 0,
        "party_size": null,
        "hero_variant": 1
    },
```


Рисунок 1.2 – Скріншот прикладу отриманих записів про гравця у json форматі

На наступному етапі здійснюється обробка отриманого JSON-файлу. Основна мета цього етапу — привести дані до табличного формату з мінімальним, але інформативним набором характеристик для подальшого аналізу. Використовується бібліотека pandas, яка дозволяє перетворити JSON у DataFrame, обробити значення та зберегти результат у CSV-файл.

Особливості обробки:

- Конвертація UNIX-мітки часу start_time у читабельний формат дати.
- Виокремлення основних статистичних показників: вбивства (kills), смерті (deaths), асисти (assists).
- Формування фінального набору даних із полями: date, hero_id, kills, deaths, assists, won.

```python
import json
import pandas as pd
from datetime import datetime, UTC

# Load JSON file
with open("dota_matches.json", "r") as file:
    matches = json.load(file)

df = pd.DataFrame(matches)
df["date"] = df["start_time"].apply(lambda x: datetime.fromtimestamp(x, UTC).strftime('%Y-%m-%d') if pd.notna(x) else
df["kills"] = df["kills"].fillna(0).astype(int)
df["deaths"] = df["deaths"].fillna(0).astype(int)
df["assists"] = df["assists"].fillna(0).astype(int)
if "radiant_win" in df.columns and "player_slot" in df.columns:
    # Determine if the player was on Radiant (0-127) or Dire (128-255)
    df["is_radiant"] = df["player_slot"] < 128
    # Determine if the player won the match
    df["won"] = df["is_radiant"] == df["radiant_win"]
    df["won"] = df["won"].astype(int)  # Convert True/False to 1/0
if "team_kills" in df.columns:
    df["kill_participation"] = (df["kills"] + df["assists"]) / df["team_kills"]
columns_to_keep = ["date", "hero_id", "kills", "deaths", "assists", "won"]
if "kill_participation" in df.columns:
    columns_to_keep.append("kill_participation")
df = df[columns_to_keep]
df.to_csv("structured_dota_matches.csv", index=False)
```

Рисунок 1.3 – Скріншот коду програми для обробки json записі та перенесення важливих даних у csv формат

Результат обробки — файл з назвою structured_dota_matches.csv, який містить всю необхідну інформацію для подальшого аналізу, зокрема кластеризації та пошуку аномальних матчів.

В таблиці 1.1 наведено невеличкий приклад вигляду кількох рядків цього CSV-файлу:

Таблиця 1.1

Приклад використаних даних про матчі

date	hero_id	kills	deaths	assists	won
2025-02-08	121	2	7	14	0
2024-12-16	64	5	8	11	0
2024-12-14	86	1	7	15	0
2024-12-14	121	2	9	8	0
2024-12-14	36	21	2	12	1
2024-12-14	121	4	7	30	1
2024-12-13	121	5	12	15	0
2024-12-13	121	5	9	22	0
2024-12-13	121	3	6	11	0
2024-12-12	14	8	9	13	0

На етапі кластеризації були використані ігри з трьох різних акаунтів з цими Steam ID: *359887171*, *447421252*, *287156880.*

На етапі аналізу аномалій був використаний лише акаунт з Steam ID: *359887171.*

Більш детальний опис датасетів представлений на наступних рисунках.

Балансний датасет акаунту 359887171 з 3290 матчами (рис. 1.4) з відсутністю пріоритетних героїв у початкових іграх і з’явившимися пріоритетними героями у середніх-останніх іграх.

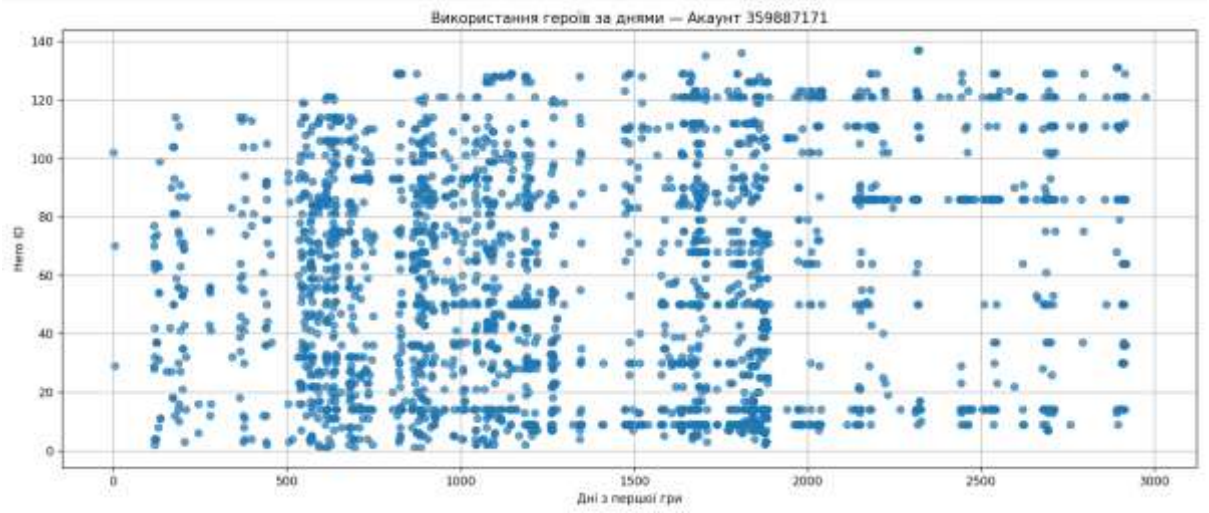


Рисунок 1.4 – Матчі з акаунту 359887171

Датасет акаунту 447421252 з 1314 матчами (рис. 1.5), у якого немає домінації певного героя.

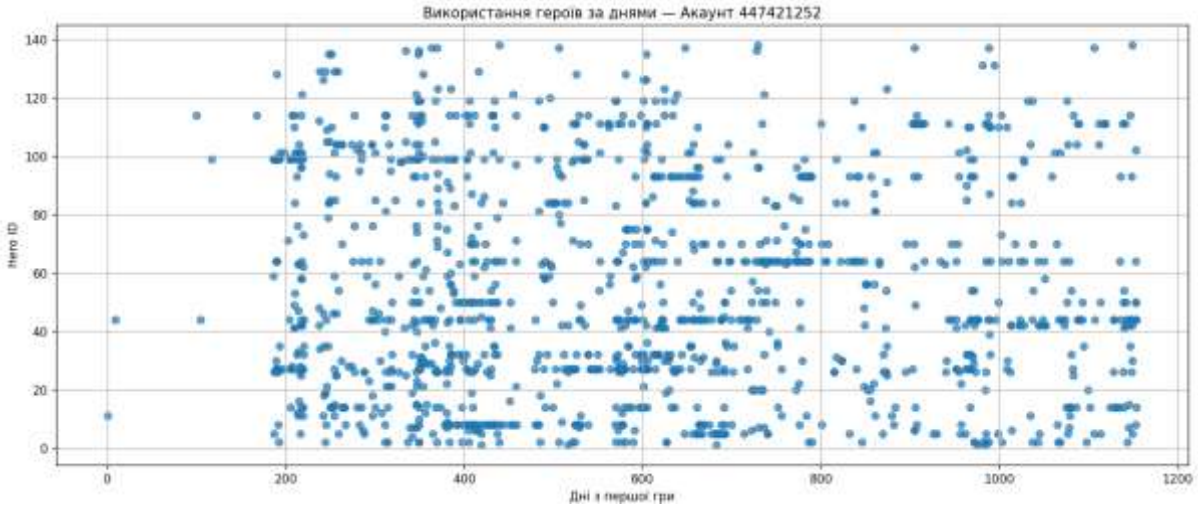


Рисунок 1.5 – Матчі з акаунту 447421252

Датасет акаунту 287156880 з 1688 матчами (рис. 1.6), з наявними домінантними героями.

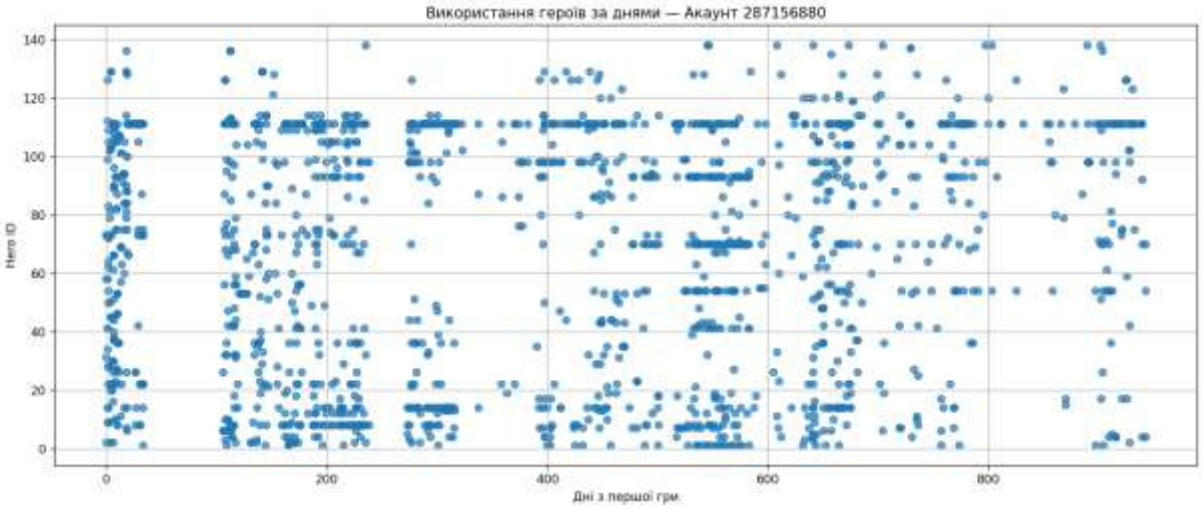


Рисунок 1.6 – Матчі з акаунту 287156880

1.3 Методи обробки матчів

1.3.1 Обробка матчів без прив’язки до дати

Цей підхід передбачає аналіз матчів гравця як лінійної послідовності ігор, без урахування календарної дати або проміжків часу між ними. Кожен матч розглядається лише у контексті своєї позиції у загальній хронології, незалежно від того, скільки часу минуло між окремими іграми.

Такий підхід дозволяє фокусуватись винятково на зміні стилю гри гравця в межах ігрових сесій, не враховуючи впливу зовнішніх факторів, таких як оновлення гри, перерви чи сезонні події. В результаті, аналіз базується на внутрішніх патернах вибору героїв, що проявляються у послідовності матчів.

У подальшій роботі, даний підхід був застосований для формування кластерів за допомогою героїв, які найчастіше використовувалися в межах певного вікна ігор, незалежно від часу їх проведення.

Цей метод також був розглянутий як альтернатива календарному підходу, однак поступився йому в контексті завдань, пов’язаних з аналізом активності у часовому розрізі.

1.3.2 Обробка матчів з прив’язкою до дати

На відміну від попереднього методу, цей підхід передбачає врахування календарної послідовності ігор, а саме — дати їх проведення. Кожен матч асоціюється з конкретним днем, і подальша обробка ґрунтується на групуванні матчів за днями.

Основна ідея полягає в тому, що ігрова активність гравця може варіювати у часі, а його стиль гри змінюється не тільки залежно від кількості ігор, але й від реального проміжку часу між ними.

У процесі такої обробки усі матчі гравця групуються за датою. Наприклад, усі ігри, зіграні 12 березня 2025 року, розглядаються як один день.

Для кожного дня будується вектор частотності використаних героїв. Ці вектори показують, наскільки часто певні герої використовувалися в той чи інший день у відсотковому відношенні.

Приклад вектору, день 724 з акаунту 359887171:

```
[0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.66666667
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.33333333
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.        ]
```

Отримані вектори, які описують розподіл героїв у межах кожного ігрового дня, використовуються як основа для обчислення подібності між ігровими днями.

Для цього застосовується косинусна подібність — метрика, що дозволяє оцінити схожість двох векторів незалежно від їхньої абсолютної величини.

$$cousine\ similarity(A, B) = \frac{A*B}{\|A\|*\|B\|}, \quad (1.2)$$

де $A * B$ – скалярний добуток векторів, $\|A\| * \|B\|$ – добуток довжин.

У даній задачі значення косинусної подібності варіюється у межах: [0 ; 1], де 1 – вектори ідентичні, 0 – вектори не мають схожостей.

Після формування всіх векторів для кожного дня будується матриця подібності розміром $N * N$, де N - кількість унікальних днів. Кожен елемент цієї матриці M_{ij} відображає косинусну подібність між векторами дня та дня j.

Ця матриця дозволяє зрозуміти, наскільки схожими були ігрові дні між собою за набором використаних героїв.

Приклад матриці M_{ij} :

```
[[1.        0.        0.        ... 0.        0.        0.        ]
 [0.        1.        0.        ... 0.        0.        0.        ]
 [0.        0.        1.        ... 0.        0.        0.        ]
 ...
 [0.        0.        0.        ... 1.        0.        0.81649658]
 [0.        0.        0.        ... 0.        1.        0.        ]
 [0.        0.        0.        ... 0.81649658 0.        1.        ]]
```

Отриману матрицю подібності було використано як основу для подальших методів кластеризації:

- Метод K-means — для фіксованої кількості кластерів, безпосередньо використовуючи вектори.
- Метод DBSCAN — з урахуванням щільності схожих днів.
- Адаптивний алгоритм кластеризації — який порівнює поточний день із попередніми та майбутніми, обираючи найбільш схожі ділянки на основі подібності векторів.

Завдяки цьому підходу вдалося згрупувати дні гравця у кластери, які враховують зміну частоти усіх героїв з часом.

1.4 Методи кластеризації

1.4.1 Метод найчастішого герою

Загальна ідея методу

Метод найчастішого героя базується на спостереженні за тим, як часто гравець використовує одного й того самого героя протягом певної кількості останніх матчів. Передбачається, що гравець може змінювати свій стиль гри або фокус на іншого героя, і ці переходи можна виявити, аналізуючи найпопулярнішого героя в нещодавніх іграх. Основна гіпотеза цього методу — домінантний герой протягом останніх матчів є ознакою стабільного стилю гри, а поява нового героя, який захоплює цю позицію, сигналізує про зміну стилю та початок нового кластера.

Побудова кластерів

Під час проходу по всіх матчах гравця, метод формує поточний кластер, додаючи до нього кожну наступну гру. Після кожного кроку перевіряється, наскільки часто певний герой зустрічається серед останніх матчів (наприклад, 20 останніх ігор). Якщо частка цього героя перевищує заданий поріг (наприклад, 50 %) і він відрізняється від поточного основного героя, то кластер завершується, і починається новий. Таким чином, перехід між кластерами ініціюється лише при появі нового домінантного героя з достатньою частотою появи.

Основні параметри

- *recent_games_window* — кількість останніх матчів, які враховуються під час визначення активного героя (наприклад, 20);
- hero_dominance_threshold — частка ігор, яку має складати найпопулярніший герой у вікні, щоб вважати, що стиль гри змінився (наприклад, 0.5 = 50%);
- min_cluster_size — мінімальна кількість ігор у кластері, щоб вважати його значущим.

Формальний опис алгоритму

Позначимо:

- впорядкований список матчів:

$$\mathrm{M} = [m_1, m_2, \dots, m_N]; \tag{1.3}$$

- герой, який використовується в матчі m_i:

$$\mathrm{H}(m_i) = [m_1, m_2, \dots, m_N]; \tag{1.4}$$

- вікно останніх w матчів:

$$W_t = [m_{t-w+1}, \dots, m_t]; \tag{1.5}$$

- кількість матчів героя h у вікні w:

$$f_h^{(W_t)}; \tag{1.6}$$

- кількість матчів найпопулярнішого героя:

$$f^{(W_t)} = \max\left(f_h^{(W_t)}\right); \tag{1.7}$$

- частка популярності героя у вікні:

$$p^{(W_t)} = \frac{f^{(W_t)}}{|W_t|}, \tag{1.8}$$

якщо:

$$p^{(W_t)} > \mathrm{hero}_{\mathrm{dominance\ threshold}} \tag{1.9}$$

і найпопопуляніший герой≠поточний герой кластеру, тоді ми змінюємо кластер і вносимо туди останні w матчів.

Переваги та обмеження

Головною перевагою методу є його простота: він не вимагає складних обчислень чи попередньої обробки даних. Однак така простота стає і основним обмеженням. Метод не враховує всю палітру героїв, які використовуються у грі. Якщо гравець стабільно грає на 2–3 героях або взагалі не має виражених пріоритетів, метод не зможе коректно виявити кластери, оскільки жоден герой не набирає достатньої частки в останніх матчах.

Крім того, кластери, які формуються, можуть бути надто великими або надто дрібними залежно від налаштувань. Якщо герой стабільно використовується протягом довгого часу, то утвориться великий кластер, який включатиме матчі з різних періодів, патчів, змін мети, що спотворює аналітику.

У зворотному випадку — коли домінуючого героя немає — гравець може залишитись з одним кластером на всі ігри, що повністю знецінює кластеризацію.

Метод також дуже чутливий до вибору гіперпараметрів — розміру вікна та порогу домінування. Зміна цих значень може радикально змінити структуру кластерів, і для кожного гравця оптимальні значення можуть бути різними.

Далі розглянемо використання алгоритму на трьох датасетах (рис.1.7, рис.1.8, рис.1.9).

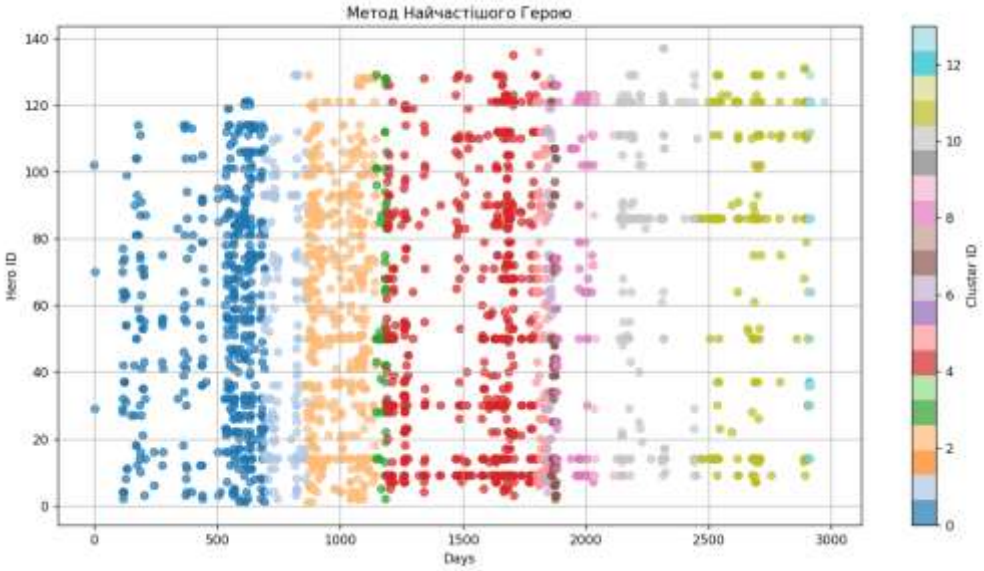


Рисунок 1.7 – Кластеризація матчів акаунту *359887171* за допомогою методу найчастішого герою

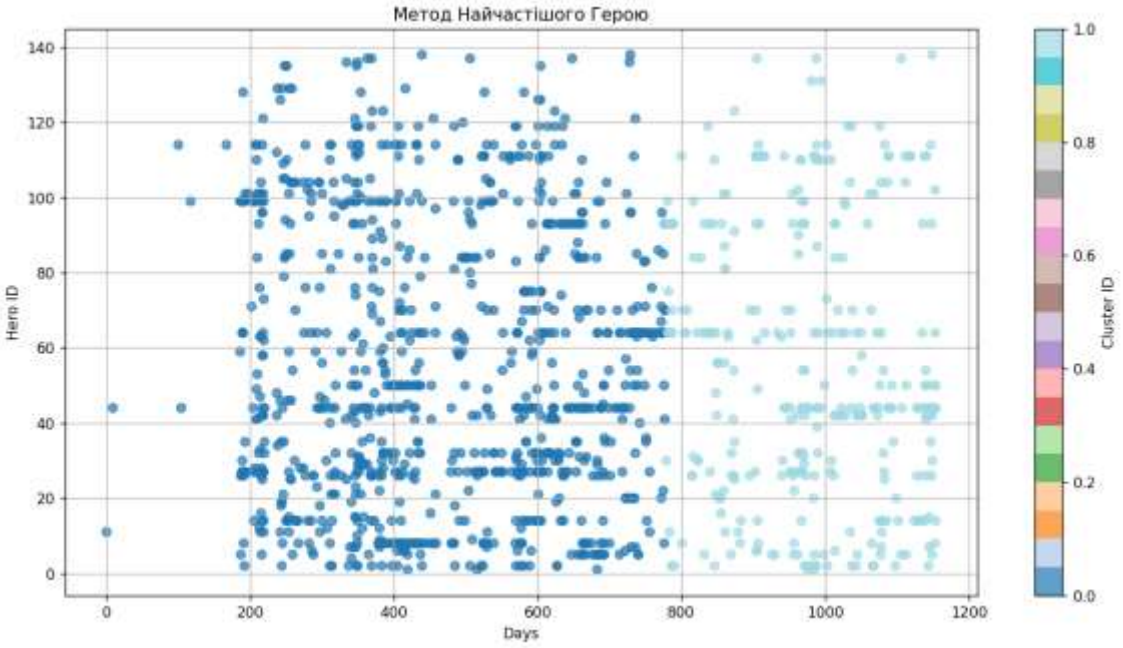


Рисунок 1.8 – Кластеризація матчів акаунту 447421252 за допомогою методу найчастішого герою

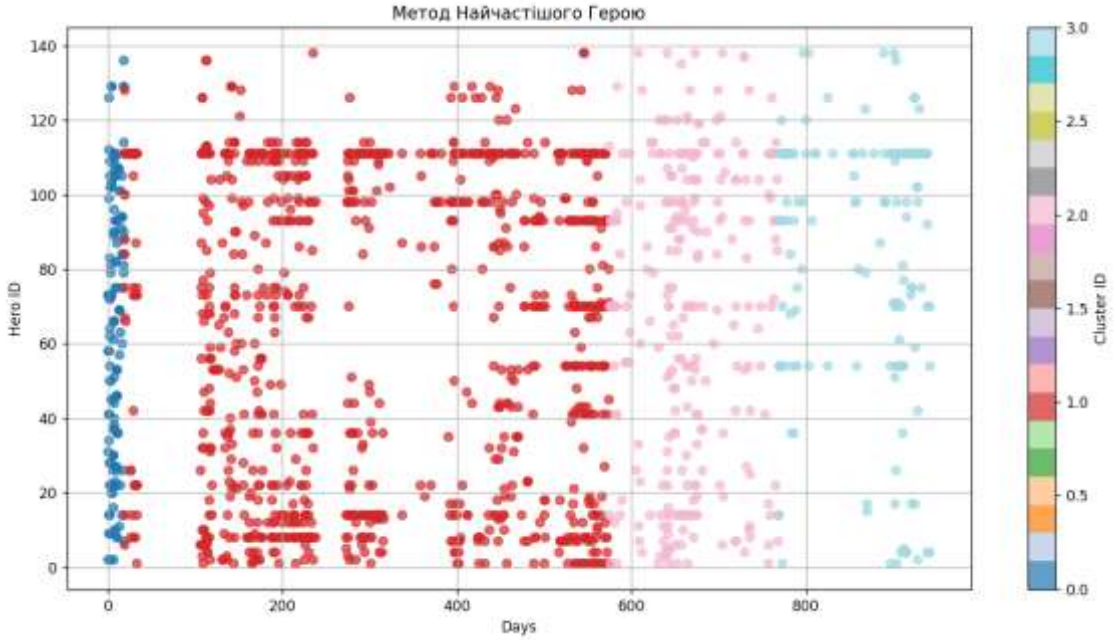


Рисунок 1.9 – Кластеризація матчів акаунту 287156880 за допомогою методу найчастішого герою

Основні недоліки даного методу

Як ми можемо побачити на практиці, алгоритм показав задовільні результати для деяких гравців. Наприклад, у випадку акаунта зі Steam ID *359887171* (рис 1.7), де стиль гри змінювався помітно і з чітко вираженими домінуючими героями в різні періоди, метод зумів побудувати змістовні кластери. Проте для акаунтів *447421252* (рис 1.8) та *287156880* (рис 1.9) він виявився малоефективним: алгоритм поділив усі матчі лише на 2–4 великі групи, що не дозволяє провести детальний аналіз стилів гри.

Це свідчить про те, що метод надто чутливий до початкових параметрів, таких як розмір вікна та поріг домінування героя. Зміна цих параметрів могла б покращити результат, однак тоді виникає потреба у їх індивідуальному налаштуванні для кожного гравця, що ускладнює масштабування методу та його автоматичне застосування.

Через перелічені обмеження метод найчастішого героя був відкинутий. Він не здатен конкретно відобразити зміну стилю гри у випадках, коли гравець варіює героїв або має більш комплексні патерни.

Натомість були розглянуті методи, які працюють з повним вектором частотності героїв та враховують зміну стилю не лише за одним героєм, а за всією поведінкою гравця в сукупності.

1.4.2 Метод K-means

Загальна ідея методу

Метод K-means є одним із найвідоміших алгоритмів кластеризації, який поділяє дані на K неперетинних кластерів на основі відстані між точками. У контексті матчів Dota 2, кожен день представлений у вигляді вектора частотності використання героїв (герой-вектор), який описує, наскільки часто певні герої використовувались у матчах, зіграних у цей день.

Таким чином, кожен день — це точка у багатовимірному просторі, де кожна вимірність відповідає одному герою.

Головна ідея полягає в тому, що схожі дні (за стилем гри) повинні належати до одного кластеру, тобто мати близькі вектори частотності героїв.

Алгоритм намагається знайти центри кластерів (центроїди), які мінімізують суму квадратів відстаней між усіма точками та їх найближчим центром.

Попередня обробка даних

Усі матчі гравця групуються за датами (1.2).

Основні параметри методу:

- $\mathrm{V} - [v_1, v_2, \ldots, v_N]$ – множина векторів частотності.
- K - задана кількість кластерів.

Формальний опис алгоритму

Алгоритм K-means виконується наступним чином:

1) ініціалізація центроїдів: випадково вибираємо K векторів як початкові центри кластерів.
2) Призначаємо точки до кластерів, для кожного v_i обчислюємо відстань до кожного центроїда:

$$d(v_i, u_k) = \|v_i - u_k\|^2 \quad (1.10)$$

3) Додаємо v_i до кластеру, до якого відстань найменша.
4) Оновлюємо центроїди для кожного кластеру як середнє арифметичне всіх векторів, що належать до нього:

$$u_k = \frac{1}{|C_k|} \sum_{v_i \in C_k} vi. \quad (1.11)$$

5) Повторюємо призначення точок та оновлення центроїдів, поки наші центроїди не перестануть змінюватися суттєво.

Приклад використання методу K-means на наших датасетах наведо на наступних рисунках:

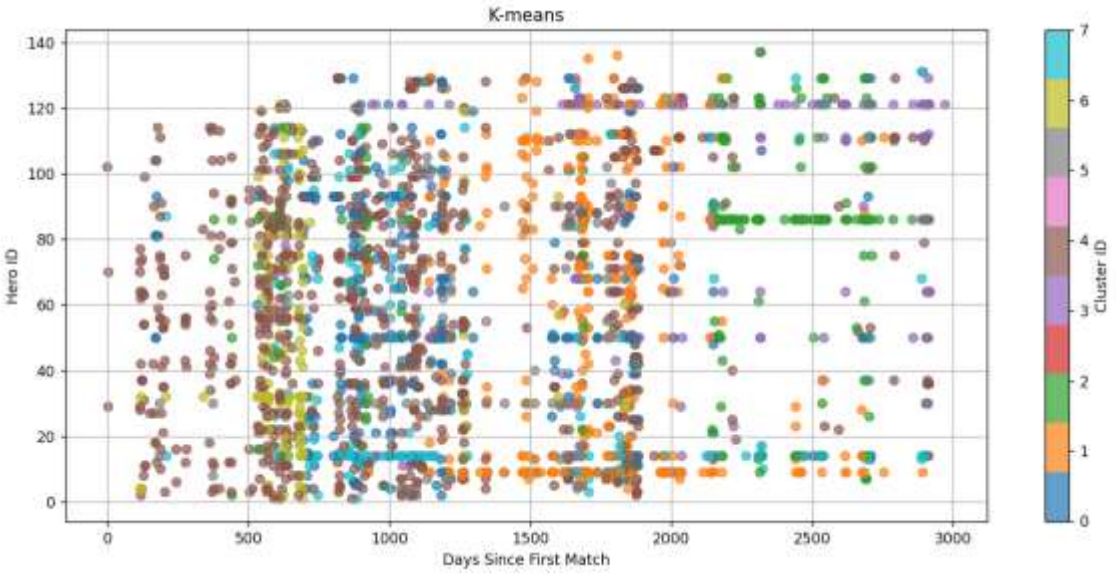


Рисунок 1.10 – Кластеризація матчів акаунту *359887171* за допомогою методу K-means

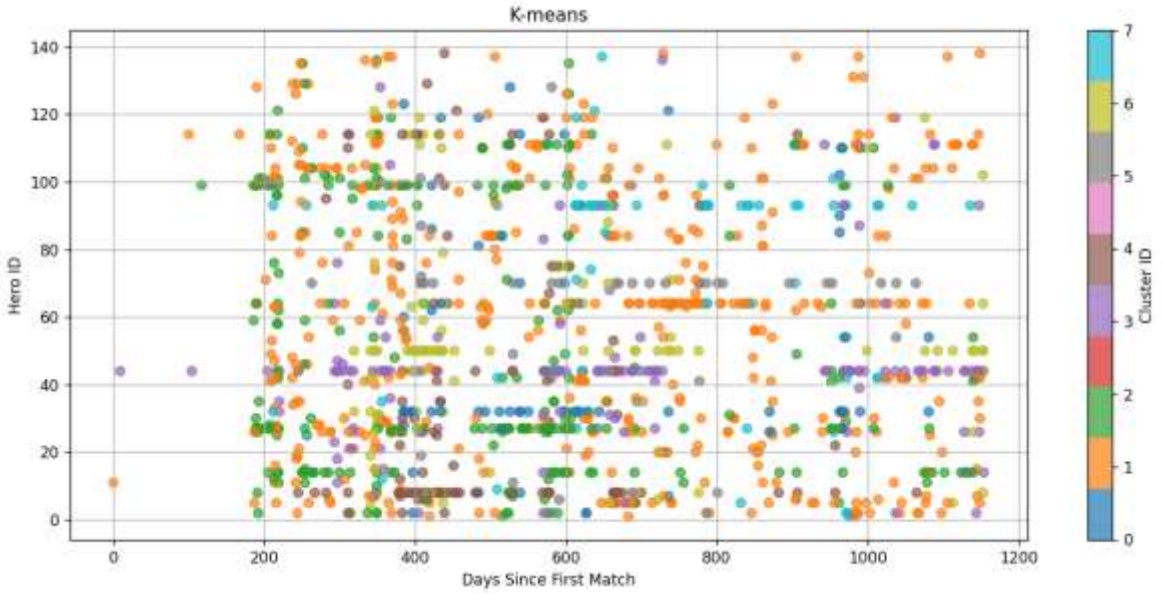


Рисунок 1.11 – Кластеризація матчів акаунту 447421252 за допомогою методу K-means

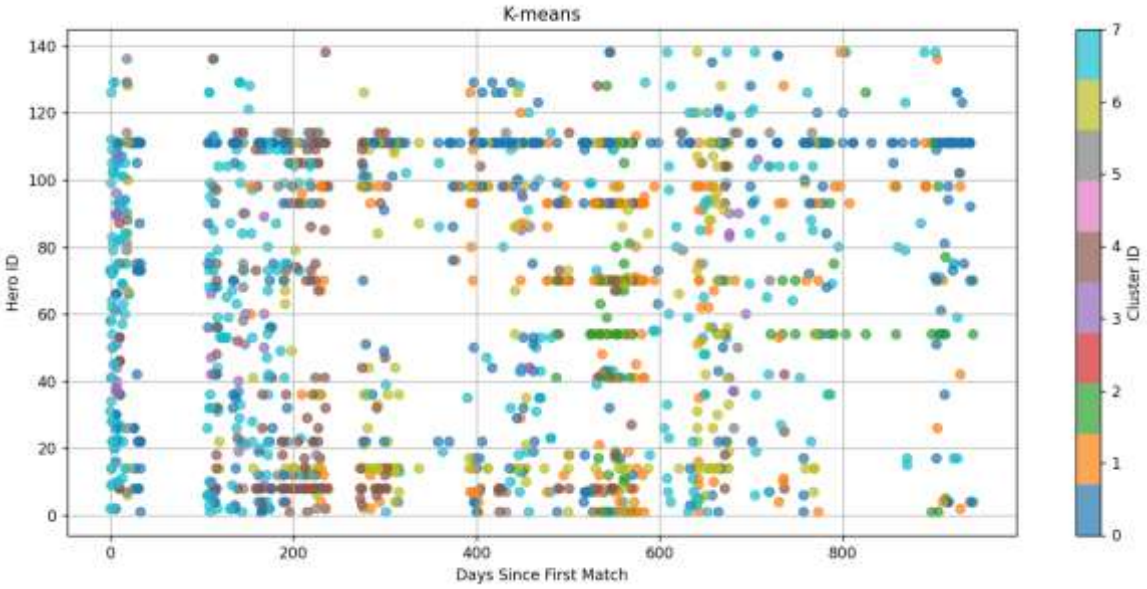


Рисунок 1.12 – Кластеризація матчів акаунту 287156880 за допомогою K-means

Основні недоліки даного методу

Перша ключова проблема — ігнорування календарного аспекту. Метод кластеризує вектори незалежно від їхньої послідовності в часі, тобто може об’єднувати стилістично подібні дні, які розділені місяцями або навіть роками (рис 1.10-1.12).

Друга важлива проблема — необхідність задавати кількість кластерів K вручну. У випадку Dota-даних ця кількість невідома наперед і може суттєво

варіюватися між гравцями. Неправильно обране K може призвести до надто загальних або надто дрібних кластерів. Крім того, неможливо гарантувати, що гравець дійсно має чітко K ігрових стилів — їх може бути більше, менше, або вони можуть змінюватись поступово, а не дискретно. Це створює додаткові труднощі для автоматичного аналізу великої кількості користувачів.

1.4.3 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Загальна ідея методу

Метод **DBSCAN** належить до класу кластеризаційних алгоритмів, які базуються на щільності даних. Його ключова концепція полягає у тому, щоб знаходити області з високою щільністю точок (в даному випадку — векторів частотності героїв за дні), розділяючи їх від областей з меншою щільністю, які вважаються шумом.

DBSCAN не вимагає попереднього знання кількості кластерів, що є його суттєвою перевагою в порівнянні з методом K-means.

Попередня обробка даних

Усі матчі гравця групуються за датами (1.2).

Основні параметри методу:

- $V = [v_1, v_2, \dots, v_N]$ – множина векторів частотності;
- ε - радіус околу;
- min_samples - мінімальна кількість точок у цьому околі для формування ядра кластера.

Формальний опис алгоритму:

1) Для кожного вектора $v_i \in V$ розглядається його ε -окол.
2) Якщо кількість точок у цьому околі ≥ min_samples, то v_i вважається точкою-ядром.
3) Усі точки в ε-околі ядра об’єднуються з ним в один кластер.
4) Процедура повторюється для кожної нової точки, яка додається в кластер, доки не буде вичерпано всі точки, які можна приєднати.

5) Точки, що не потрапили до жодного кластера, вважаються шумом (noise points).

Для розрахунку подібності між точками ми використовували косинусну подібність як метрику відстані (1.2).

Оскільки DBSCAN вимагає метрику відстані, ми конвертували косинусну подібність у відстань як:

$$cousine\ distance(A, B) = 1 - \ cousine\ similarity(A, B) \quad (1.12)$$

Після використання алгоритму на наших датасетах отримали наступні результати використання методу DBSCAN (рис. 1.13, рис. 1.14, рис. 1.15).

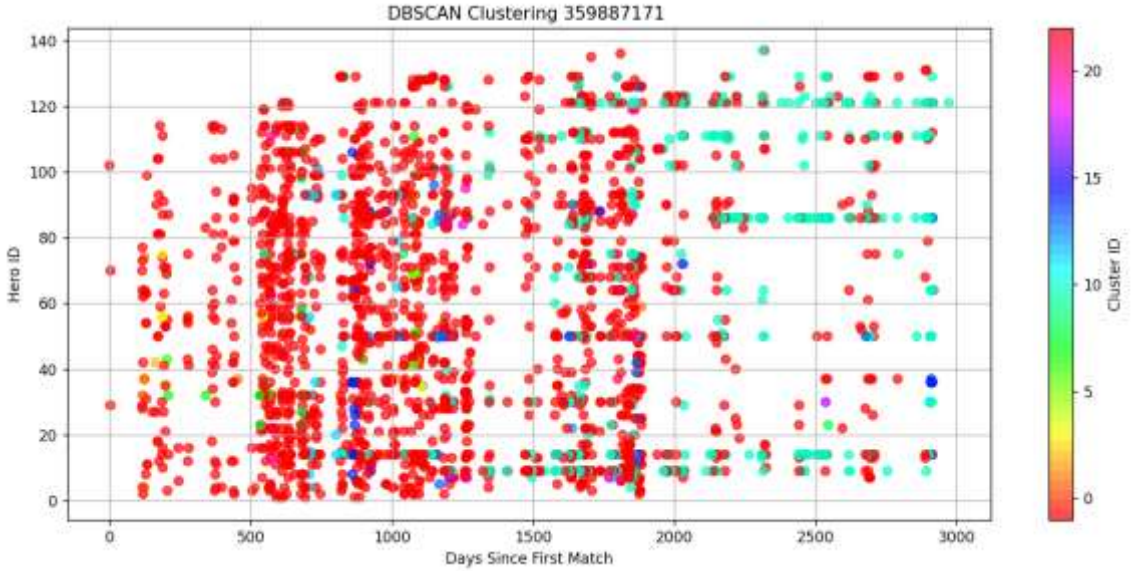


Рисунок 1.13 – Кластеризація матчів акаунту 359887171 за допомогою методу DBSCAN з параметром ε = 0.15

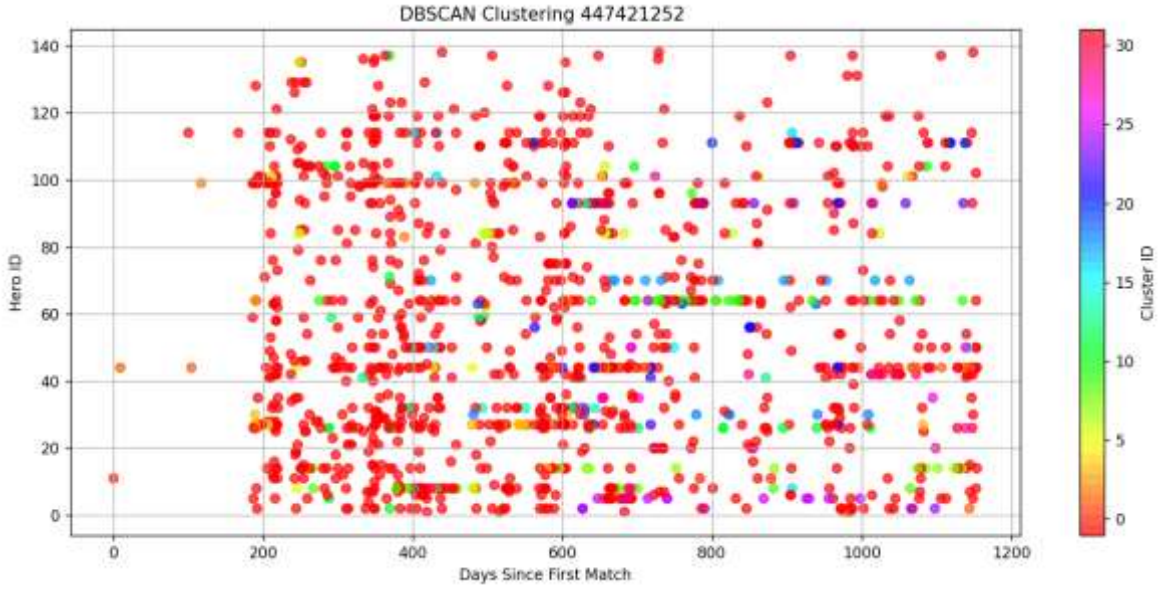


Рисунок 1.14 – Кластеризація матчів акаунту 447421252 за допомогою методу DBSCAN з параметром ε = 0.15

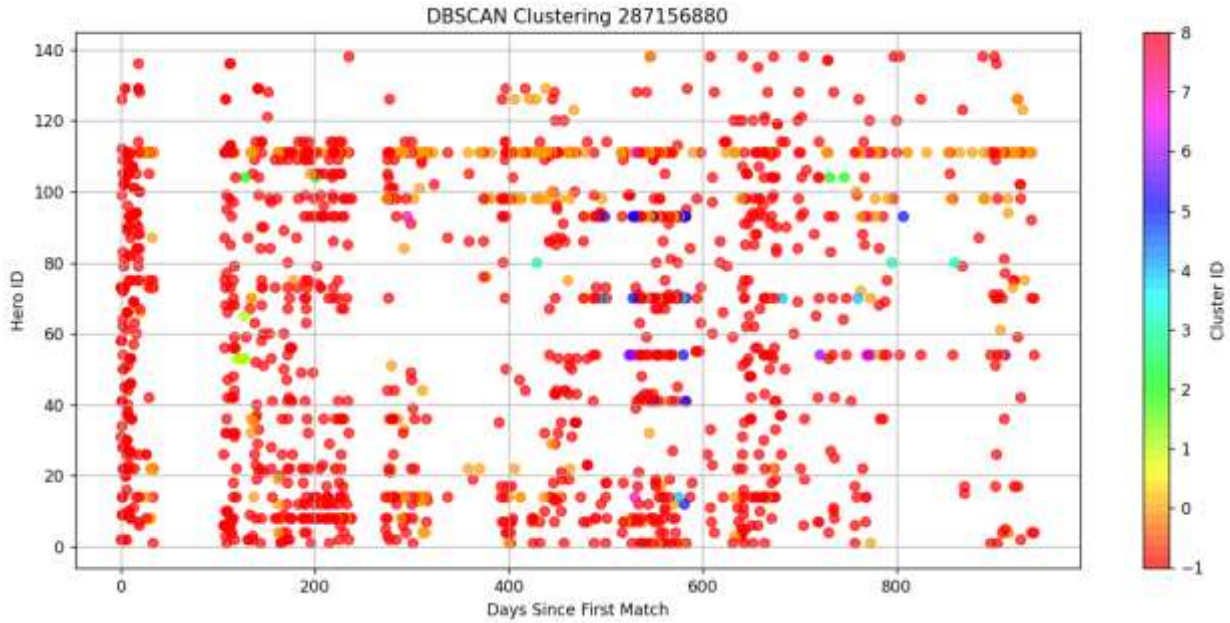


Рисунок 1.15 – Кластеризація матчів акаунту 287156880 за допомогою методу DBSCAN з параметром ε = 0.15

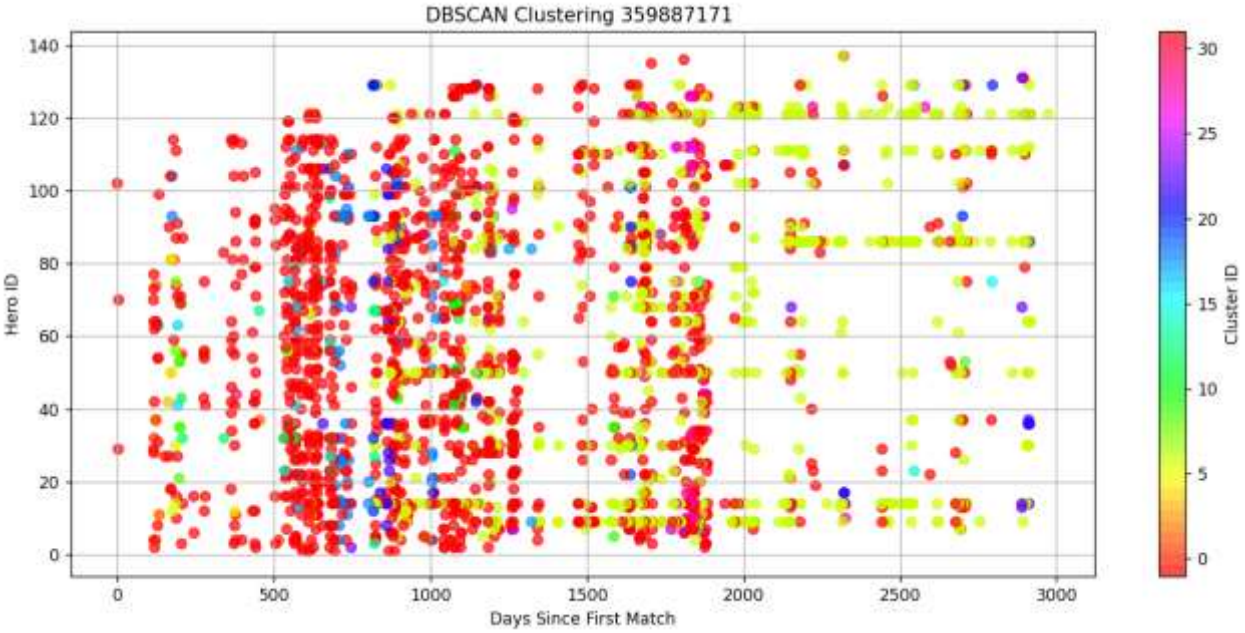


Рисунок 1.16 – Кластеризація матчів акаунту 359887171 за допомогою методу DBSCAN з параметром ε = 0.25

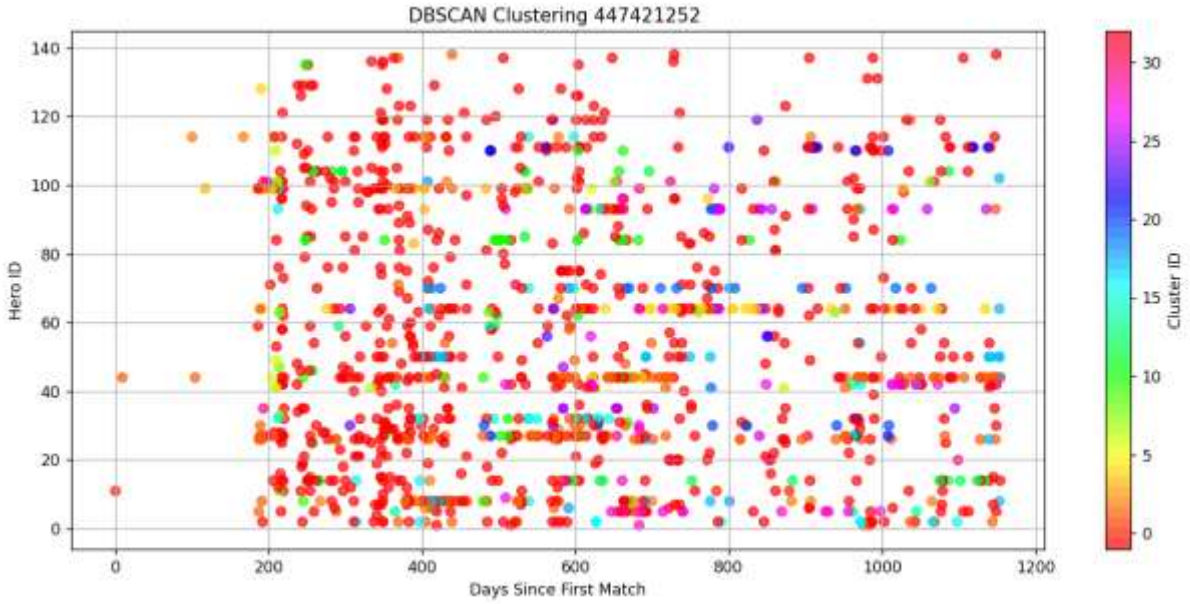


Рисунок 1.17 – Кластеризація матчів акаунту 447421252 за допомогою методу DBSCAN з параметром ε = 0.25

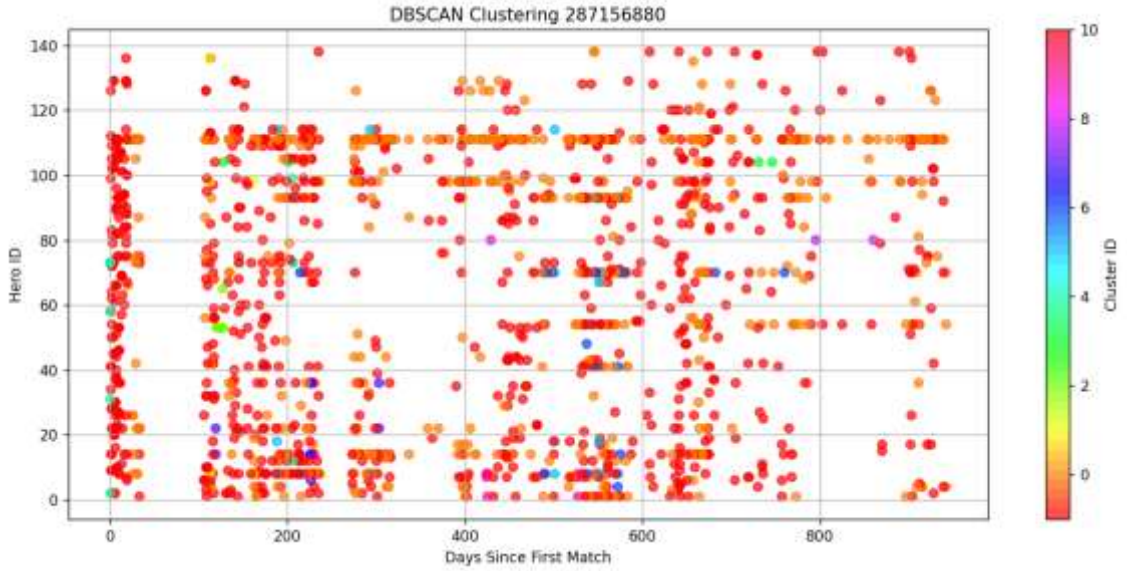


Рисунок 1.18 – Кластеризація матчів акаунту 287156880 за допомогою методу DBSCAN з параметром ε = 0.25

Основні недоліки даного методу

Першою суттєвою проблемою методу DBSCAN є надмірна чутливість до параметрів: радіуса пошуку ε та мінімальної кількості точок у кластері min_samples. Невдало підібрані параметри можуть або об’єднувати занадто різнорідні дні в один кластер, або, навпаки, позначати більшість днів як шум, не включаючи їх у жоден кластер. Оскільки DBSCAN не призначений для даних з неоднорідною щільністю, цей недолік особливо виражений у Dota-матчах, де стиль гри може змінюватися поступово, і кластерні межі не завжди мають чіткі щільні зони.

Друга проблема — ігнорування хронологічного порядку ігор. Подібно до K-means, DBSCAN аналізує вектори на основі просторової подібності у векторному просторі, не беручи до уваги, коли саме ці матчі були зіграні (рис 1.13-1.18). Це є проблемою, оскільки ми маємо такі ситуації: v_1, v_2, v_3 – вхідні параметри

- Використання формули 1.2 для комбінації пар параметрів

$$cousine\ distance(v_1, v_2) = 0$$

$$cousine\ distance(v_1, v_3) = 0.5$$

$$cousine\ distance(v_2, v_3) = 0.5$$

Якщо ε = 0.5, ми б отримали усі три вектори у одному кластері, що не є великою проблемою у малому маштабі, але якщо кожен вектор має доступ до усіх інших векторів, це може створювати кластери, що не мають ніякого відношення один до одного.

1.5 Висновки до розділу 1

У розділі було описано повний процес: від отримання матчів через OpenDota API до їх обробки у структурований формат. Дані конвертовано у вектори частотності героїв по днях, що дозволило оцінювати стиль гри гравця в часовому розрізі. Було протестовано кілька методів кластеризації (метод найчастішого героя, K-means, DBSCAN), які виявились недостатньо ефективними через такі обмеження: ігнорування хронології, залежність від гіперпараметрів та слабка здатність враховувати поступові зміни стилю гри.

РОЗДІЛ 2 СПЕЦІАЛЬНИЙ

2.1 Спеціальний адаптивний метод кластеризації на основі косинусної подібності

Загальна ідея

У багатьох ігрових аналітичних задачах важливо не лише знати, які герої були обрані гравцем, але й як змінювався його стиль гри протягом часу. Особливо в іграх на кшталт Dota 2, де мета (певні дії, які використовуються більшістю гравців для досягнення перемоги на даний момент часу), патчі, баланс персонажів, особисті вподобання гравця або його роль у команді можуть змінюватися щодня. Для цього потрібно не просто кластеризувати дані — потрібно зробити це з урахуванням часової послідовності матчів.

У класичних підходах (наприклад, K-Means), кожен день або матч розглядається як окрема точка у багатовимірному просторі, і кластери формуються поза контекстом послідовності. Проте такий підхід ігнорує важливу динаміку — еволюцію стилю гри з плином часу.

Саме тому в даному підході основна ідея полягає у поступовому проходженні по кожному дню (тобто по кожному вектору частотності пік героїв), починаючи з першого.

Інтуїція формування кластерів

Замість того, щоб заздалегідь визначати кількість кластерів або довжину кожного, метод поступово аналізує, чи продовжує гравець грати в тому ж стилі, чи стиль змінився. Кожен новий день порівнюється:

- з усередненим стилем попереднього кластеру;
- з можливими схожими днями у майбутньому.

Це дає змогу:

- уникнути включення днів до кластера, якщо гравець вже почав грати по-іншому;
- не створювати новий кластер без підтвердження того, що стиль справді змінився;

- відловлювати точки перелому в поведінці гравця природним способом.

Перевага поступового аналізу

Класичні алгоритми працюють над набором точок і потім “заднім числом” групують їх. У нашому ж випадку ми діємо в реальному порядку подій:

- День за днем ми оцінюємо схожість поточного стилю до попереднього.
- Якщо стиль стабільний, день додається до поточного кластеру.
- Якщо стиль починає змінюватися, ми перевіряємо майбутні дні — чи підтверджують вони зміну.
- Якщо підтверджують — кластер завершується, і починається новий.

Таким чином, ми реалізуємо адаптивну логіку, яка дозволяє не лише класифікувати гру гравця, а й відслідковувати зміни у його ігровій поведінці у реальному часі.

Можна уявити це як моніторинг серії подій у хронологічному порядку, де ми постійно вирішуємо: “Чи це ще та сама ситуація, чи вже нова?”. Замість того, щоб просто поділити все на частини за схожістю, ми шукаємо природні точки зміни, які логічно випливають з ігрових даних.

Вхідні параметри:

- ✓ $\theta \in [0\,;1]$ - поріг косинусної подібності (наприклад 0.5);
- ✓ $w \in N$ - розмір вікна наступних ігор, що розглядається у методі (наприклад 20);
- ✓ $m \in N$ - мінімальна кількість днів, які мають порівнюватися з нинішнім (наприклад 10);

Формальний опис алгоритму:

1) $C_0 = \{0\}$ – перший кластер містить v_0, $C_0 = 0$.

2) Виконуємо основний цикл поки $i < N$.

3) Обчислюємо майбутній інтервал:

$$F_i = \{v_j | i + m \le j \le min(i + w, n - 1), cos(v_i, v_j) > \theta\}. \qquad (2.1)$$

4) Якщо множина F_i не порожня, визначаємо індекс останнього схожого дня:

$$\mathrm{j}^* = \max\{i \in F_i\}. \tag{2.2}$$

5) Обчислюємо середню схожість до поточного кластеру.

$$S_{cluster} = \frac{1}{|C_c|}\sum_{k \in C_c} \cos(v_i, v_k). \tag{2.3}$$

6) Обчислюємо середню схожість до майбутніх днів.

$$S_{future} = \frac{1}{|F_i|}\sum_{j \in F_i} \cos(v_i, v_j) \tag{2.4}$$

7) Якщо $S_{cluster} \geq S_{future}$:

- додаємо v_i до поточного кластеру C_c;
- збільшуємо крок $i = i + 1$.

8) Якщо $S_{cluster} < S_{future}$:

- закриваємо кластер C_c;
- починаємо новий кластер $C_{c+1} = \{v_i, v_{i+1}, \dots, v_{j^*}\}$;
- присвоюємо новий номер $c = c + 1$;
- переходимо до позиції $i = j^* + 1$.

Після завершення алгоритму отримаємо множину кластерів $\{C_1, C_2, \dots, C_k\}$, де кожен кластер $C_k \subset \{0, \dots, n-1\}$ є впорядкованою послідовністю днів зі схожим стилем гри.

Використання алгоритму на датасетах наведено на наступних рисунках.

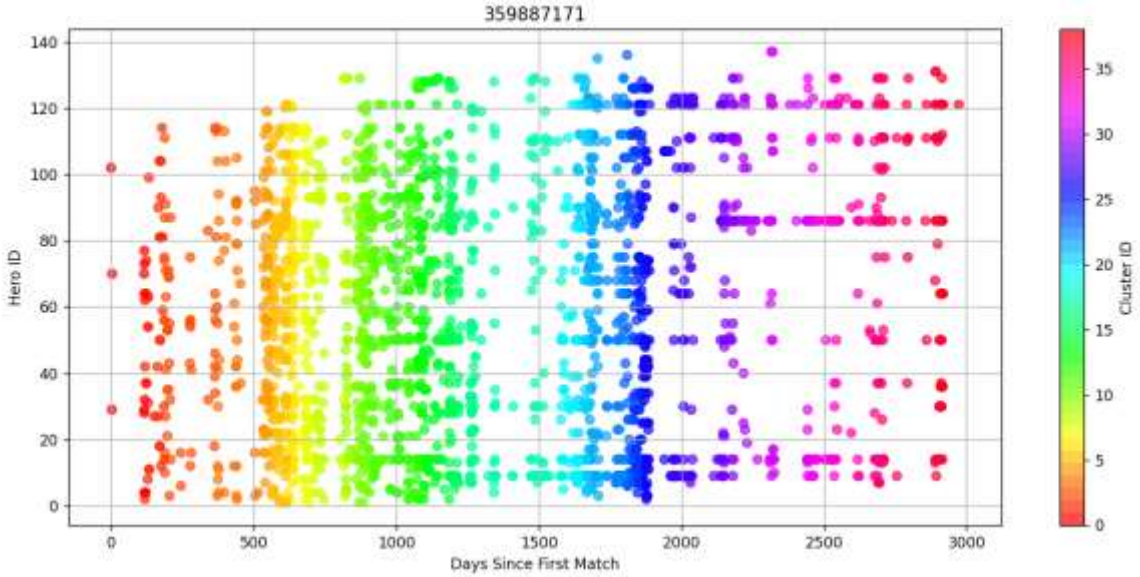


Рисунок 2.1 – Кластеризація матчів акаунту 359887171 за допомогою розробленого спеціального адаптивного методу

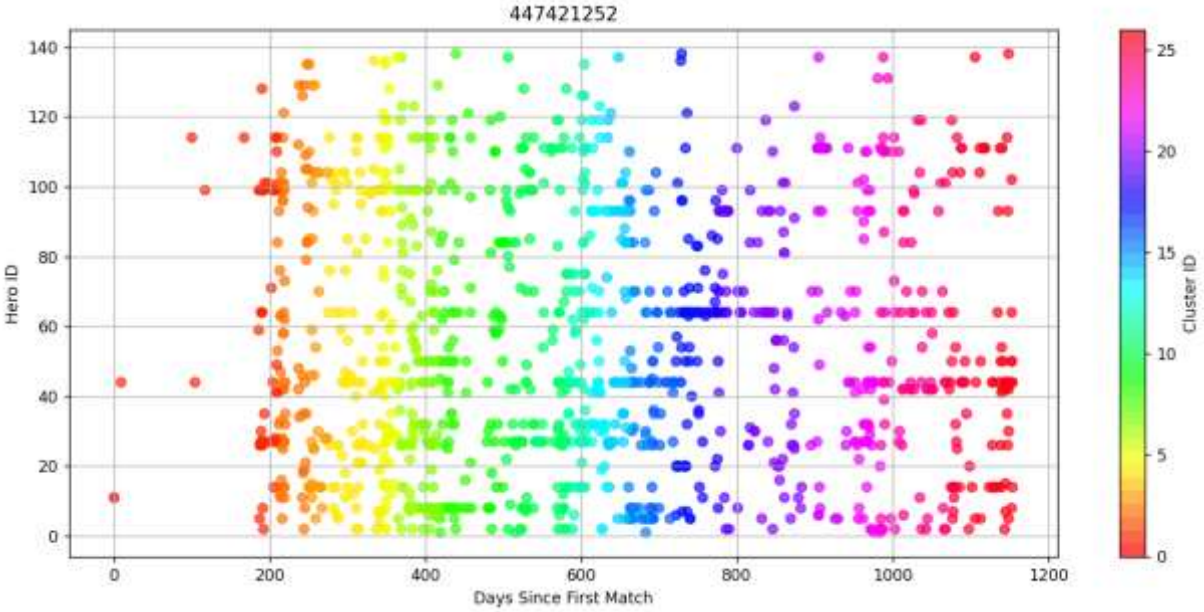


Рисунок 2.2 - Кластеризація матчів акаунту 447421252 за допомогою за допомогою розробленого спеціального адаптивного методу

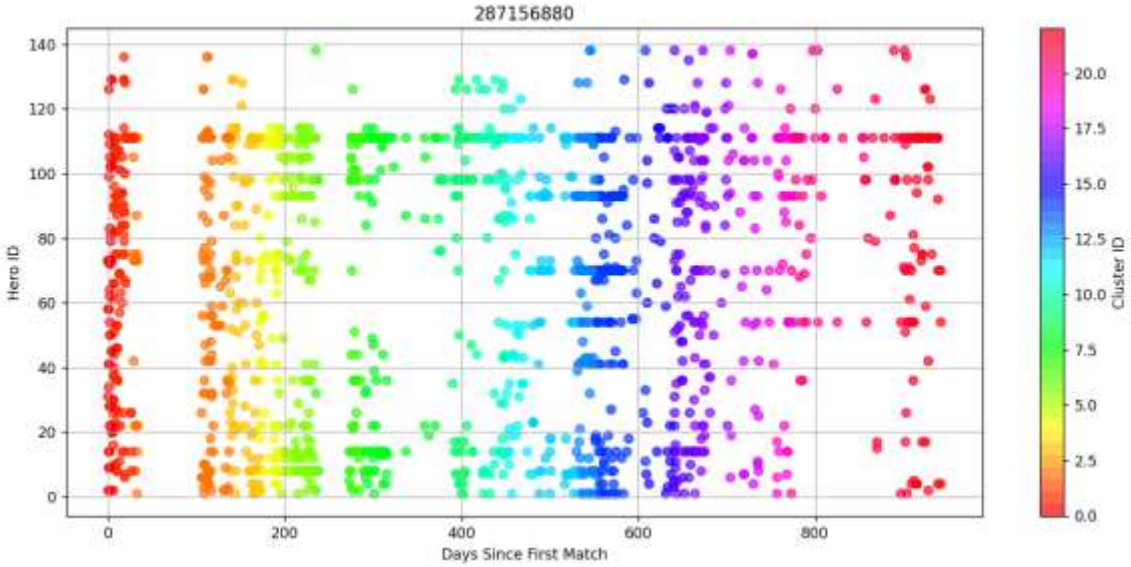


Рисунок 2.3 - Кластеризація матчів акаунту 287156880 за допомогою розробленого спеціального адаптивного методу

Як ми можемо побачити, кластери повільно перетікають з прогресією днів.

Слід зазначити, що середнє значення днів у кластері є біля 20 для кожного з датасетів.

Тож можемо зробити висновок головний висновок - *розроблений алгоритм з однаковими параметрами працює на різних датасетах.*

Топ 5 персонажів відносно кластерів 0-12 з датасету 359887171:

Cluster 0 (Top 5 Heroes):
hero_id
73 10.909091
9 7.272727
20 7.272727
22 5.454545
102 5.454545
Name: count, dtype: float64

Cluster 1 (Top 5 Heroes):
hero_id
111 21.641791
107 3.731343
26 3.731343
94 2.985075
66 2.985075
Name: count, dtype: float64

Cluster 2 (Top 5 Heroes):
hero_id
111 26.351351
53 4.729730
73 4.054054
22 4.054054
32 3.378378
Name: count, dtype: float64

Cluster 3 (Top 5 Heroes):
hero_id
111 11.764706
8 7.058824
114 4.705882
2 3.529412
60 3.529412
Name: count, dtype: float64

Cluster 4 (Top 5 Heroes):
hero_id
111 14.925373
8 10.447761
109 7.462687
56 5.970149
22 5.970149
Name: count, dtype: float64

Cluster 5 (Top 5 Heroes):
hero_id
111 23.404255
93 10.638298
8 8.510638
105 8.510638
14 6.382979
Name: count, dtype: float64

Cluster 6 (Top 5 Heroes):
hero_id
8 18.478261
111 11.956522
14 7.608696
36 7.608696
93 5.434783
Name: count, dtype: float64

Cluster 7 (Top 5 Heroes):
hero_id
111 20.895522
98 14.925373
14 11.940299
36 7.462687
8 7.462687
Name: count, dtype: float64

Cluster 8 (Top 5 Heroes):
hero_id
111 33.333333
14 18.888889
8 5.555556
13 4.444444
22 4.444444
Name: count, dtype: float64

Cluster 9 (Top 5 Heroes):
hero_id
98 21.538462
111 21.538462
14 7.692308
7 6.153846
93 4.615385
Name: count, dtype: float64

Cluster 10 (Top 5 Heroes):
hero_id
111 27.692308
98 6.153846
126 4.615385
7 4.615385
14 4.615385
Name: count, dtype: float64

Cluster 11 (Top 5 Heroes):
hero_id
111 25.000000
35 7.692308
14 5.769231
53 5.769231
43 3.846154
Name: count, dtype: float64

Cluster 12 (Top 5 Heroes):

```
hero_id
111    16.666667
70     10.606061
93     10.606061
7       9.090909
54      7.575758
Name: count, dtype: float64
```

Як ми можемо побачити, найбільш популярні герої у кластерах 1-3 займають значно більший відсоток ніж ті, що нижче. Це показує, що кластери мають сенс.

Також бажано відділити наступні результати використання алгоритму:

```
Cluster 7 (Top 5 Heroes):
hero_id
111    20.895522
98     14.925373
14     11.940299
36      7.462687
8       7.462687
Name: count, dtype: float64

Cluster 8 (Top 5 Heroes):
hero_id
111    33.333333
14     18.888889
8       5.555556
13      4.444444
22      4.444444
Name: count, dtype: float64

Cluster 9 (Top 5 Heroes):
hero_id
111    33.333333
14     18.888889
8       5.555556
13      4.444444
22      4.444444
Name: count, dtype: float64
```

Всі наведені вище дані говорять про те, що головний герой 111 не змінився, але другий герой 98 зник, а замість нього піднявся 14.

Тобто алгоритм працює відмінно, враховуючи зміни у різних героях, а не тільки в основного героя. Це говорить про можливість його використання для різних над\лаштувань у різних акаунтах.

2.2 Використання розробленого методу аналізу аномалій

Загальна ідея

Аномалію у цьому контексті вважаємо неприроднє завищений коефіцієнт KDA (1.1), що вирізняється на фоні інших граючих днів на тому самому герої.

Ціль нашого методу - виявити кластери днів де KDA несподівано високе і відсутні близькі середні значення у сусідніх кластерах.

Етап 1: Обчислення кластерної KDA-статистики для кожного героя

Для кожного героя вичислили глобальне середнє і стандартне відхилення значень KDA (1.1):

$$\mu_h = E[KDA_h] \tag{2.5}$$

$$\sigma_h = \sqrt{V[KDA_h]} \tag{2.6}$$

Етап 2: Пошук кластерів, які виділяються.

Для кожного героя h та кожного кластера C_i, який брав участь у грі, обчислено:

1) $\overline{KDA_{C_i}}$ - середнє значення KDA у кластері.

2) $\overline{KDA_{rest}}$ - середнє значення KDA у інших кластерах $\left(C_j \neq C_i\right)$.

До кластера застосовується умова сумнівної активності:

$$\overline{KDA_{C_i}} > E[\overline{KDA_{rest}}] + 2 * S[\overline{KDA_{rest}}] \tag{2.7}$$

Етап 3: Додаткова фільтрація за сусіднім кластерним середнім

Якщо ігра в підозрілому кластері виявилась аномалією, тобто $KDA > \mu_h + 2\sigma_h$, тоді ми перевіряємо чи є сусідні (± 2) кластери з цим же героєм, у яких:

$$\left|\overline{KDA_{nearby}} - \mu_h\right| < 0.5 * \sigma_h \tag{2.8}$$

Якщо такі кластери є, гра не фіксується як сумнівна.

Формування фінального списку

Усі ігри, що потрапили у кластери зі статистично значимо високим KDA (1.1) і де сусідні кластери не приземлюють його середнє значення, позначаються як фінальні підозрілі ігри і записуються у CSV-файл final_suspicious_matches.csv.

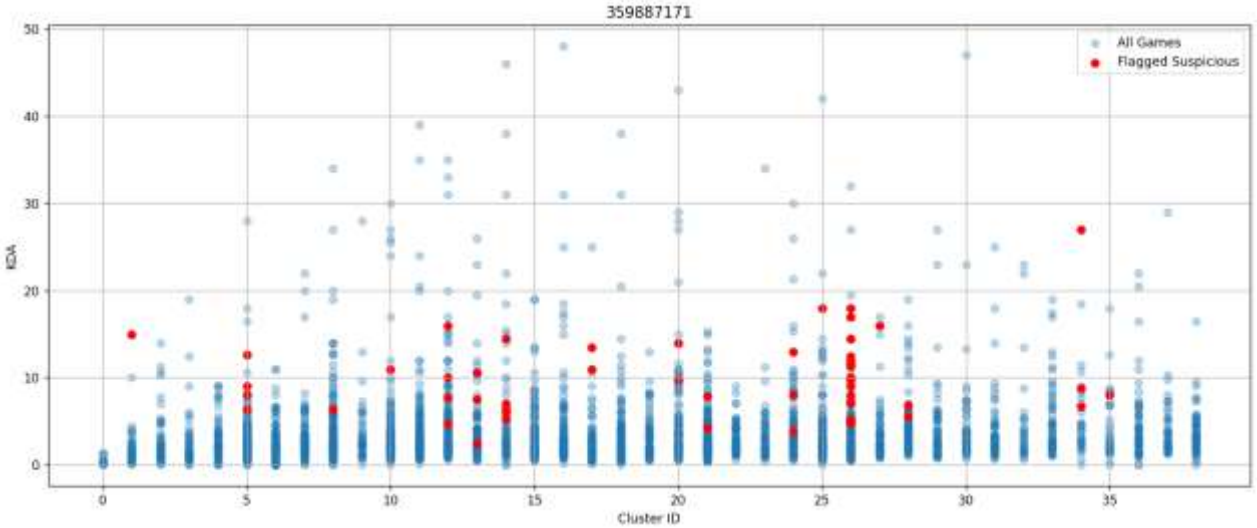


Рисунок 2.4 – Аномальні матчі акаунту 359887171 з кожного кластеру

Приклад даних, які отримані з файлу final_suspicious_matches.csv і для розуміння наданої інформації позначено:

- зелені – правильно;
- жовті – тяжко визначити;
- червоні – неправильно.

Таблиця 2.1

Аномальні матчі знайдені алгоритмом аналізу аномалій

date	hero_id	kills	deaths	assists
2024-04-19	110	3	4	29
2024-02-21	110	5	3	15
2024-01-28	22	11	1	16
2023-12-07	129	17	5	27
2022-07-23	123	20	6	13
2022-07-11	123	10	4	17
2022-06-19	30	11	2	21
2022-02-16	74	8	2	10
2022-02-15	74	9	0	9
2022-02-15	74	23	3	6
2022-02-15	74	16	4	12
2022-02-14	74	22	8	16
2022-02-14	74	12	5	14
2022-02-13	74	16	6	13
2022-02-13	74	20	6	11
2022-02-13	74	16	2	13
2022-02-12	74	15	2	8
2022-02-12	74	17	3	12
2022-02-09	74	13	2	11
2022-02-07	74	10	2	15
2022-02-07	74	18	6	11
2022-02-07	74	16	3	19
2022-02-06	74	15	2	19
2022-02-06	39	18	3	16
2022-02-06	74	15	4	14
2022-02-06	74	14	2	15
2022-02-05	6	16	3	14
2022-01-29	74	8	2	8
2022-01-21	34	25	2	11
2022-01-06	113	15	2	11
2022-01-05	29	5	3	19
2021-12-25	105	14	6	9
2021-08-21	53	5	4	12
2021-07-22	91	4	6	43
2021-07-11	9	5	3	37
2021-06-21	121	4	3	25
2020-06-19	39	11	3	22
2020-06-09	77	12	2	15
2020-01-08	11	12	2	17
2019-12-22	128	12	5	20
2019-12-21	85	7	7	30
2019-12-21	85	11	6	25
2019-12-20	128	8	4	20
2019-12-09	69	5	2	10

Продовження таблиці 2.1

2019-12-07	43	13	3	19
2019-11-29	69	11	9	11
2019-10-29	2	21	7	12
2019-10-05	65	8	3	22
2019-09-21	20	3	3	20
2019-09-17	68	4	2	28
2019-04-26	36	16	3	17
2019-03-02	20	11	5	21
2018-08-22	106	20	6	18
2018-08-02	81	31	3	7
2018-07-30	56	4	1	4
2018-07-15	13	2	1	7
2017-06-09	50	4	2	26

Тобто після використання запропонованого алгоритму пошуку аномалій, ми отримали наступні результати:

- Правильні аномальні матчі: 66%.
- Тяжко визначити: 10%.
- Неправильно визначені: 23%.

Слід зазначити, що алгоритм зміг знайти серію аномальних матчів з 2022-02-05 до 2022-02-16.

2.3Висновки до розділу 2

В другому розділі кваліфікаційної роботи розроблено алгоритм кластеризації ігрових сесій на основі розподілу героїв за періодами, реалізовано і виконано виявлення кластерів із підвищеними показниками KDA, а також зроблена перевірка наявності подібних ігор у сусідніх кластерах з метою верифікації аномалій.

Слід зазначити, що розроблений адаптивний метод забезпечив найкращі результати кластеризації, оскільки враховав послідовність ігор у часі, динаміку стилю гри та схожість із майбутніми періодами.

ВИСНОВОК

У межах цієї кваліфікаційної роботи було реалізовано комплексний підхід до аналізу поведінки гравців у грі Dota 2, заснований на щоденній активності у виборі героїв.

Основною метою кваліфікаційної роботи було виявлення змін у стилі гри з наступною кластеризацією ігор, а також виявлення потенційно аномальних періодів з непропорційно високими показниками ефективності.

У першому розділі проведено опис та порівняння кількох методів кластеризації даних, включно з K-Means, DBSCAN та спеціально розробленим адаптивним методом на основі косинусної подібності. Кожен метод має свої переваги та недоліки: зокрема, K-Means вимагає наперед заданої кількості кластерів і не враховує хронологію, DBSCAN — чутливий до параметрів щільності.

У другому розділі був розроблений адаптивний метод, що забезпечив найкращі результати кластирізації, оскільки враховує послідовність ігор у часі, динаміку стилю гри та схожість із майбутніми періодами.

Додатково було розроблено механізм виявлення аномалій на основі показника KDA. Метод дозволяє виявити не просто окремі "вдалі" матчі, а саме кластери, у яких результати суттєво перевищують звичну ефективність гравця, і при цьому не підтверджуються середніми показниками у суміжних періодах. Це дозволяє з високою ймовірністю виявляти нетипову поведінку, включаючи підозру на зміну гравця, буст, або інші неприродні фактори.

У результаті дослідження отримано:

- ✓ гнучку методику кластеризації щоденних ігрових векторів;
- ✓ систему виявлення KDA-аномалій з фільтрацією за контекстом;
- ✓ деталізовану сегментацію ігрових сесій, що дозволяє оцінити зміну стилю гри протягом часу;

Практична цінність. Запропонований підхід може бути масштабовано для аналізу великої кількості користувачів та використаний у системах автоматичної модерації, виявлення підозрілих профілів або як інструмент дослідницького аналізу поведінкових патернів у кіберспорті.

Апробація результатів кваліфікаційної роботи здійснена на II (VIII) Міжнародній науково-практичній конференції здобувачів вищої освіти і молодих учених «Інформаційні технології: теорія і практика» (2 – 4 квітня 2025 р., м. Запоріжжя – Харків – Дніпро, Україна). Отримані наукові результати відображено у публікації [2].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кваліфікаційна робота бакалавра [Електронний ресурс] : методичні рекомендації для здобувачів ступеня бакалавра освітньо-професійної програми «Системний аналіз» зі спеціальності 124 Системний аналіз / уклад.: Т.А. Желдак, Т.В. Хом'як, А.В. Малієнко ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2025. – 32 с. url: https://ir.nmu.org.ua/handle/123456789/170863
2. Ambartsumian, S. & Shevchenko, Y. (2025). TIME-SERIES CATEGORICAL DATA CLUSTERING. Information technologies: theory and practice: Abstracts of the VIII (II) International Internet Conference of Higher Education Applicants and Young Scientists (Zaporizhzhia-Kharkiv-Dnipro, April 2-4, 2025), [Electronic resource] Electronic data. – Zaporizhzhia: National University "Zaporizhzhia Polytechnic", p. 107-110.
3. https://docs.opendota.com/
4. https://pandas.pydata.org/docs/
5. Коряшкіна, Л. С., Станіна, О. Д., & Шевченко, Ю. О. (2024). Практикум з диференційних рівнянь. https://ir.nmu.org.ua/handle/123456789/167658
6. https://allenai.org/blog/specter2-adapting-scientific-document-embeddings-to-multiple-fields-and-task-formats-c95686c06567
7. Introduction to Python [Електронний ресурс] // Google Developers. – Режим доступу: https://developers.google.com/edu/python/introduction
8. Ambartsumian, S., & Yuliia, S. (2026). TIME-SERIES CATEGORICAL DATA CLUSTERING. https://ir.nmu.org.ua/handle/123456789/172323
9. Машинне навчання [Електронний ресурс] : конспект лекцій для здобувачів ступеня магістра зі спеціальності 124 Системний аналіз / уклад.: Т.А. Желдак, О.Б. Владико ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2024. – 162 с. Режим доступу: https://ir.nmu.org.ua/handle/123456789/167978 (дата звернення: 01.04.2025)
10. Коряшкіна Л.С. Практикум з диференційних рівнянь [Електроний ресурс] : навч. посіб. / Л.С. Коряшкіна, О.Д. Станіна, Ю.О. Шевченко; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка» - Дніпро : НТУ «ДП», 2024 – 178 с. Режим доступу: https://ir.nmu.org.ua/handle/123456789/167658

11. Шевченко, Ю.О., 2022. Обробка і аналіз даних з використанням електронних таблиць. Частина І «Обробка даних». Режим доступу: https://ir.nmu.org.ua/handle/123456789/162623 (дата звернення 08.06.2025)
12. Коряшкіна Л.С. Практикум за курсом «Методи оптимізації та дослідження операцій». Частина І. Дослідження операцій: навч. посіб. / Л.С. Коряшкіна, С.А. Ус / М-во освіти і науки України; Нац. техн. ун-т «Дніпровська політехніка». – Д.: НТУ «ДП», 2020. – 182 с. Режим доступу: https://ir.nmu.org.ua/handle/123456789/166157 (дата звернення: 01.10.2024).
13. Виробнича практика [Електронний ресурс] : методичні рекомендації для здобувачів ступеня магістра освітньо-професійної програми «Системний аналіз» спеціальності 124 Системний аналіз / уклад.: Т.А. Желдак, О.Д. Станіна, С.А. Ус ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2024. – 24 с. Режим доступу: https://ir.nmu.org.ua/handle/123456789/167906 (дата звернення: 01.04.2025)
14. Коряшкіна, Л., Малієнко, А., Станіна, О., Шевченко, Ю., Кодола, Я. (2025). Системний аналіз та оптимальний вибір комплексу заходів для підвищення безпеки на підприємстві. Information Technology: Computer Science, Software Engineering and Cyber Security, 72–80, doi: https://doi.org/10.32782/IT/2025-2-7
15. Системний аналіз [Електронний ресурс] : методичні рекомендації до виконання практичних робіт для здобувачів ступеня бакалавра зі спеціальності 124 Системний аналіз / уклад.: А.В. Малієнко, О.Б. Владико, С.В. Козир, Д.М. Гаранжа ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2024. – 82с. Режим доступу: https://ir.nmu.org.ua/handle/123456789/167652 (дата звернення: 01.05.2025)
16. Математична економіка [Електронний ресурс] : методичні рекомендації до практичних занять для здобувачів в ступеня бакалавра спеціальності 124 Системний аналіз (F4 Системний аналіз та наука про дані) / уклад.: Л.С. Коряшкіна, С.В. Козир, М.М. Одновол ; М-во освіти і науки України, Нац. техн. ун-т «Дніпровська політехніка». – Дніпро : НТУ «ДП», 2025. – 66 с. (дата звернення: 01.05.2025)

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

№ з/п	Позначення	Найменування	Кількість аркушів	Примітки
1				
2		Документація		
3				
4	САУ.КР.25.1.ПЗ	Пояснювальна записка	45	Формат А4
5				
6	САУ.КР.25.1.ДМ	Демонстраційний матеріал		Презентація на CD–R
7				
8	САУ.КР.25.1.КР	Копія роботи	1	Диск CD–R
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				

					САУ.КР.25.19.ДА.ПЗ.			
Змін.	Аркуш	№ докум.	Підпис	Дата				
Розроб.		Амбарцумян С.П.			**Матеріали кваліфікаційної роботи**	Літ.	Аркуш	Аркушів
К. розд.		Шевченко Ю.О.						
Керівн.		Шевченко Ю.О.				НТУ «ДП», 12; 124–21–2		
Н.контр.		Хом’як Т.В.						
Зав. каф.		Желдак Т.А.						

ДОДАТОК Б. Відгук

на кваліфікаційну роботу бакалавра

здобувача вищої освіти групи *124 – 21 – 2*
спеціальності *124 Системний аналіз*

Тема кваліфікаційної роботи: Аналіз і обробка ключових показників аккаунту гравця у комп'ютерній грі для пошуку аномальних даних
Обсяг кваліфікаційної роботи 45 стор.
Мета кваліфікаційної роботи: кластирізація матчів глявця для подальшого аналізу статистики та пошуку аномальних ігор.
Актуальність теми полягає у створенні інструменту, який дозволяє гравцям DOTA 2 проаналізувати свою ігрову статистику в розрізі часових періодів, визначити найуспішніші стратегії гри, а також виявити потенційні ситуації передачі акаунту іншим користувачам, що може вплинути на чесність змагань.
Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності здобувача спеціальності 124 Системний аналіз, оскільки вона передбачає глибоке вивчення, структурування та вдосконалення методів аналізу даних, що є ключовими завданнями системного аналітика.
Виконані в кваліфікаційній роботі завдання відповідають вимогам ступеня бакалавра. Оригінальність наукових рішень полягає в аналізі існуючих методів кластерізації, а також в створенні нових, що більш підходять до приведеної проблеми.
Практичне значення результатів кваліфікаційної роботи полягає в отриманні значної кількості аномальних матчів завдяки використанню створеного підходу, що показує ефективність даного алгоритму.
Висновки підтверджують можливість використання результатів роботи для аналізу різних акаунтів у грі Дота 2
Оформлення пояснювальної записки та демонстраційного матеріалу до неї виконано згідно з вимогами.
Роботу виконано самостійно, відповідно до завдання та у повному обсязі (*в разі невідповідності – вказати*)
У роботі відзначено такі недоліки:

Кваліфікаційна робота в цілому заслуговує оцінки: відмінно (98).

З урахуванням висловлених зауважень автор (не) заслуговує присвоєння освітньої кваліфікації «бакалавр з системного аналізу».

Керівник кваліфікаційної роботи бакалавра,
асистент кафедри САУ _____ / Юлія ШЕВЧЕНКО

ДОДАТОК В. Участь у конференції

Міністерство освіти і науки України
Національний університет
«Запорізька політехніка»
Національний технічний університет
«Дніпровська політехніка»
Харківський національний університет
міського господарства імені О.М. Бекетова
ГО «Системні дослідження»
ГО МДЦВЕ
Esslingen University of Applied Sciences
University of Koblenz
Cardiff University
Kırklareli University
Universidad Politécnica de Madrid

INFORMATION TECHNOLOGIES:
THEORY AND PRACTICE

II (VIII) International
Scientific and Practical Conference
of Students and Young Scientists

ABSTRACTS OF REPORTS

April 2 - 4, 2025

Zaporizhzhia-Kharkiv-Dnipro
Ukraine

SECTION 2. INTELLIGENT COMPUTER SYSTEMS

UDC 681.51:519.876.5 Ambartsumian S.[1], Shevchenko Y.[2]

TIME-SERIES CATEGORICAL DATA CLUSTERING

Topic relevance. Clustering is a widely used technique in data analysis, primarily applied to numerical datasets where data points can be compared using distance metrics. However, applying traditional clustering methods to categorical and time-series data presents several challenges.

One major issue is the **lack of a natural distance metric**. Numerical clustering algorithms, such as k-means, rely on Euclidean distance to group similar points, but categorical values like character names or roles do not have an inherent numerical relationship. Methods like k-modes adapt k-means by using the mode instead of centroids, but they still fail to capture sequence patterns over time

Another problem is the **temporal nature of the data**. Standard clustering techniques group data based on static attributes, whereas time-series clustering must account for evolving trends. For instance, in gaming analytics, a player's character choices change over time, requiring an algorithm that detects shifts rather than fixed categories

Hierarchical clustering offer potential solutions, but they are computationally expensive and not optimized for large datasets with categorical values.

Therefore, a more efficient approach is needed to handle categorical time-series data while maintaining computational feasibility.

Problem investigation. To address these challenges, we decided to create a **sequential clustering algorithm** which designed specifically for categorical time-series data.

This algorithm groups data points based on evolving trends rather than static similarities, making it well-suited for applications such as player behavior analysis in games.

Key Parameters and Settings:

1) *Recent Games Window* (recent_games_window):

- defines the number of past matches considered when determining a player's dominant playstyle;
- a larger window smooths short-term fluctuations but may delay

[1] Student at the Department of System Analysis and Control, Dnipro University of Technology, Dnipro, Ukraine

[2] Assistant Lecturer at the Department of System Analysis and Control, Dnipro University of Technology, Dnipro, Ukraine

detection of playstyle shifts.

2) *Hero Dominance Threshold* (hero_dominance_threshold):

- represents the minimum proportion of recent matches where a hero must appear to be considered dominant;
- for example, if set to 0.5, the hero must be used in at least 50% of the recent games to trigger a cluster change.

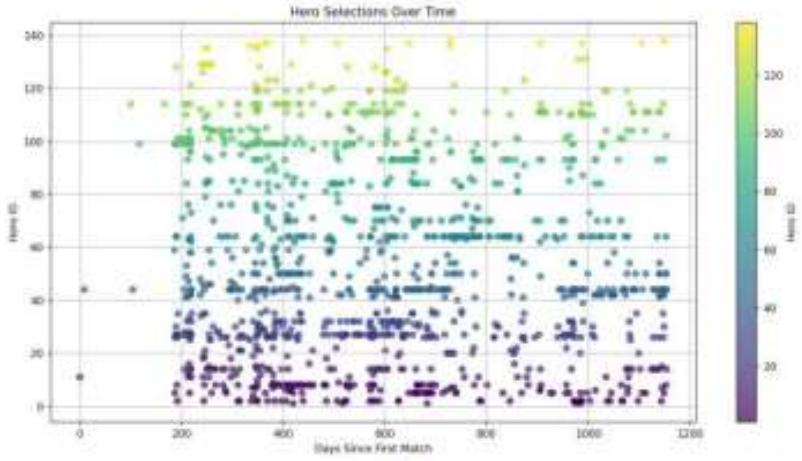
3) *Minimum Cluster Size* (min_cluster_size):

- prevents the formation of overly small clusters that may result from brief variations in hero selection;
- ensures clusters represent stable playstyle trends rather than one-off anomalies.

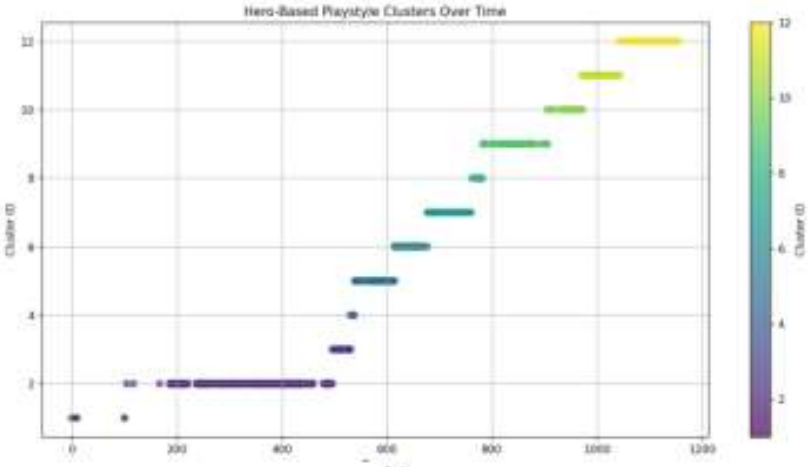
*How the **Algorithm for Categorical Time-Series Clustering** works:*

1. **Reprocessing:** Convert timestamps into a numerical format (days since first match) and sort matches chronologically.
2. **Tracking Hero Trends**: Maintain a sliding window of recent_games_window matches to detect dominant heroes.
3. **Cluster Formation**: If a hero dominates within this window beyond hero_dominance_threshold, a new cluster is created.
4. **Cluster Refinement**: Small clusters are merged or discarded based on min_cluster_size to ensure meaningful segmentation.

On the Figure 1 (*a,b,c*) you can example of usage algorithm, where 1315 games; recent_games_window = 10; hero_dominance_threshold = 0.5; min_cluster_size = 0.



a)



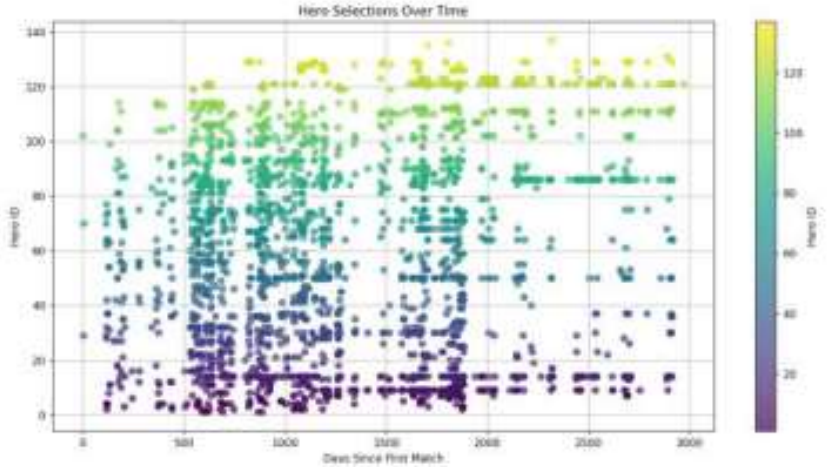
b)

	Cluster Number	Total Games	Most Popular Hero 1	Hero 1 Percentage
0	1	3	11.0	33.33%
1	2	520	8.0	7.12%
2	3	57	27.0	15.79%
3	4	10	99.0	50.00%
4	5	148	27.0	20.27%
5	6	113	44.0	13.27%
6	7	109	64.0	16.51%
7	8	26	64.0	30.77%
8	9	89	93.0	15.73%
9	10	63	111.0	14.29%
10	11	72	44.0	16.67%
11	12	104	44.0	17.31%

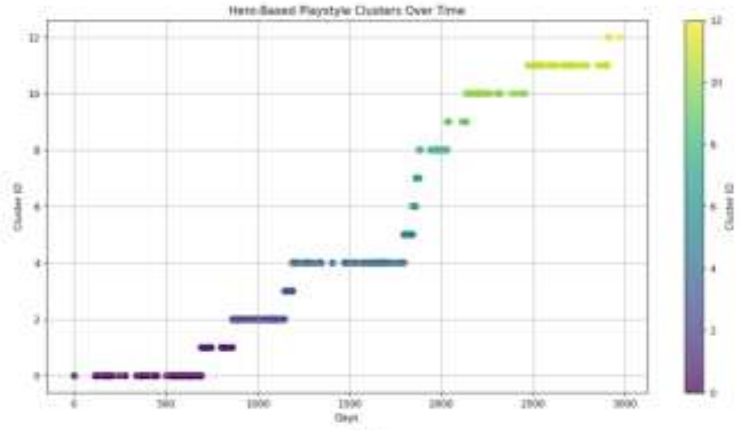
c)

Figure 1 - Example of usage algorithm with 1315 games

On the Figure 2 (*a, b, c*) you can example of usage algorithm with 91 games; recent_games_window = 20; hero_dominance_threshold = 0.5; min_cluster_size = 20.



a)



b)

	Cluster Number	Total Games	Most Popular Hero 1	Hero 1 Percentage
0	0	572	32.0	5.07%
1	1	228	93.0	31.14%
2	2	573	50.0	6.98%
3	3	132	58.0	44.70%
4	4	710	9.0	21.13%
5	5	147	64.0	38.88%
6	6	122	9.0	38.52%
7	7	123	50.0	20.33%
8	8	124	9.0	20.16%
9	9	58	121.0	60.34%
10	10	200	86.0	40.50%
11	11	218	86.0	25.69%
12	12	26	121.0	50.00%

c)

Figure 2 - Example of usage algorithm with 3291 games

Practical significance. Unlike traditional clustering methods, it dynamically adapts to changes in categorical sequences, making it a practical solution for time-series categorical data processing.

REFERENCES

1. Aghabozorgi S., Shirkhorshidi A., Wah T. Time-series clustering. A DecadeReview Information systems. 2015. Vol. 53. P. 16-38.

2. Pandas : веб-сайт. URL: https://pandas.pydata.org/docs/ (дата звернення: 25.02.2025).

3. Matplotlib : веб-сайт. URL: https://matplotlib.org/stable/users/index.html (дата звернення: 25.02.2025).

4. Scikit-learn : веб-сайт. URL: https://scikit-learn.org/stable/user_guide.html (дата звернення: 25.02.2025).

5. Коряшкіна Л. С., Станіна О. Д., Шевченко Ю. О. Практикум з диференційних рівнянь. 2024. URL: