КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

Звіт до лабораторної роботи №2 з курсу «Основи Data Mining»

Студента групи ТТП-41
Маркова Максима Юрійовича

викладач: Терещенко Ярослав Васильович

Київ-2025

Анотація

В цій лабораторній роботі було реалізовано сегментацію клієнтів за допомогою алгоритму K-means для виявлення груп з подібними характеристиками. Основні етапи роботи:

- Попередня обробка даних, включаючи завантаження, очищення від пропусків, вибір релевантних ознак та нормалізацію даних для рівного внеску кожної ознаки в обчислення відстаней.
- Дослідження даних, включаючи візуалізацію розподілу ключових змінних, таких як вік, річний дохід та рівень витрат.
- Реалізація кластеризації за допомогою алгоритму K-means, з вибором оптимальної кількості кластерів за допомогою Elbow Method та Silhouette Analysis (з порівнянням методів) та аналізом характеристик кожного кластера.
- Оцінка якості кластерів за допомогою Silhouette Score та інтерпретація результатів у контексті бізнес-задач.
- Візуалізація результатів кластеризації з використанням методів зниження розмірності, таких як PCA, для побудови двовимірного графіку, що демонструє розподіл клієнтів по кластерах.

Для реалізації було обрано мову програмування Python через наявність потужних бібліотек для аналізу даних та візуалізації. Було використано наступні бібліотеки:

- Pandas — для обробки та маніпулювання даними.
- Scikit-learn — для реалізації алгоритму K-means та нормалізації даних.
- Matplotlib і Seaborn — для побудови графіків та візуалізації результатів.

- Scikit-learn (PCA) — для зниження розмірності та візуалізації кластерів.

Отримані результати можуть бути використані для аналізу поведінки клієнтів, визначення цільових груп та розробки стратегій персоналізованого маркетингу.

Вступ

Тема роботи: «Сегментація клієнтів».

Мета роботи – провести сегментацію клієнтів за допомогою алгоритму K-means для виявлення груп з подібними характеристиками та візуалізувати отримані результати за допомогою сучасних методів аналізу даних.

Лабораторна робота складається з наступних етапів:

1. **Попередня обробка даних**
    - Завантаження та очищення даних, включаючи обробку пропусків та видалення аномальних значень.
    - Вибір релевантних ознак для подальшого аналізу та нормалізація даних для забезпечення рівного внеску кожної ознаки в кластеризацію.
2. **Дослідження даних**
    - Аналіз розподілу ключових характеристик клієнтів, таких як вік, річний дохід, витрати тощо.
    - Візуалізація зв’язків між різними змінними для розуміння потенційних патернів.
3. **Кластеризація клієнтів**
    - Використання методу K-means для сегментації клієнтів.
    - Визначення оптимальної кількості кластерів за допомогою Elbow Method та Silhouette Analysis.
4. **Оцінка якості кластерів**
    - Оцінка якості кластеризації за допомогою показників, таких як Silhouette Score, а також інтерпретація результатів у контексті бізнес-проблеми.

5. **Візуалізація результатів**
    - Використання методів зниження розмірності, таких як PCA, для візуалізації клієнтів у просторі кластерів.
    - Побудова графіків для візуалізації характеристик кожного кластера та розподілу клієнтів по групах.

Лабораторна робота дозволяє застосувати техніки кластеризації для аналізу поведінки клієнтів, що може бути корисним для персоналізованого маркетингу, створення рекомендаційних систем та оптимізації бізнес-стратегій.

1. Теоретична частина та аналіз алгоритму

1.1 Алгоритм K-Means

Алгоритм K-Means — популярний метод кластеризації, — впорядкування множини об'єктів у порівняно однорідні групи. Винайдений в 1950-х роках математиком Гуґо Штайнгаузом і майже одночасно Стюартом Ллойдом. Особливу популярність отримав після виходу роботи МакКвіна (1967).

Мета методу — розділити n спостережень на k кластерів, так щоб кожне спостереження належало до кластера з найближчим до нього середнім значенням. Метод базується на мінімізації суми квадратів відстаней між кожним спостереженням та центром його кластера, тобто функції $\sum_{i=1}^{N} d(x_i, m_j(x_i))^2$, де d — метрика, x_i — і-ий об'єкт даних, а $m_j(x_i)$ — центр кластера, якому на j-ій ітерації приписаний елемент x_i.

1.2 Опис алгоритму K-Means

Маємо масив спостережень (об'єктів), кожен з яких має певні значення за рядом ознак. Відповідно до цих значень об'єкт розташовується у багатовимірному просторі.

1. Дослідник визначає кількість кластерів k, що необхідно утворити
2. Випадковим чином обирається k спостережень, які на цьому кроці вважаються центрами кластерів
3. Кожне спостереження «приписується» до одного з k кластерів — того, відстань до якого найкоротша
4. Розраховується новий центр кожного кластера як елемент, ознаки якого розраховуються як середнє арифметичне ознак об'єктів, що входять у цей кластер
5. Відбувається така кількість ітерацій (повторюються кроки 3-4), поки кластерні центри стануть стійкими (тобто при кожній ітерації в

кожен кластер потрапляють одні й ті самі об'єкти), дисперсія всередині кластера буде мінімізована, а між кластерами — максимізована.

Вибір кількості кластерів робиться на основі дослідницької гіпотези. Якщо її немає, то рекомендують спочатку створити 2 кластери, далі 3, 4, 5, порівнюючи отримані результати.

1.3 Переваги та недоліки алгоритму K-Means

Головні переваги методу k-середніх — його простота та швидкість виконання. Метод k-середніх більш зручний для кластеризації великої кількості спостережень, ніж метод ієрархічного кластерного аналізу (у якому дендограми стають перевантаженими і втрачають наочність).

Одним із недоліків простого методу є порушення умови зв'язності елементів одного кластера, тому розвиваються різні модифікації методу, а також його нечіткі аналоги, у яких на першій стадії алгоритму допускається приналежність одного елемента множини до декількох кластерів (із різним ступенем приналежності).

Попри очевидні переваги методу, він має суттєві недоліки:

- Результат класифікації сильно залежить від початкових позицій кластерних центрів
- Алгоритм чутливий до викидів, які можуть викривлювати середнє
- Кількість кластерів має бути заздалегідь визначена дослідником

1.4 Elbow Method

У кластеризації K-Means ми починаємо з випадкової ініціалізації k кластерів і ітеративного коригування цих кластерів, поки вони не стабілізуються в точці рівноваги. Однак перш ніж ми зможемо це зробити, нам потрібно вирішити, скільки кластерів (k) ми повинні використовувати.

Метод ліктя допомагає нам знайти це оптимальне значення k. Ось як це працює:

1. Ми “пробігаємо” діапазон значень k, як правило, від 1 до n (де n — параметр, який ми вибираємо).
2. Для кожного k ми обчислюємо суму квадратів у межах кластера (WCSS; Within-Cluster Sum of Squares).

WCSS вимірює, наскільки добре точки даних згруповані навколо відповідних центроїдів. Він визначається як сума квадратів відстаней між кожною точкою та її центроїдом кластера:

$WCSS = \sum_{i=1}^{k} \sum_{j=1}^{n_i} d(x_j^{(i)}, c_i)^2$, де $d(x_j^{(i)}, c_i)^2$ представляє відстань між j-ю точкою даних $x_j^{(i)}$ у кластері i та центроїд c_i цього кластера.

Метод ліктя працює за наступною схемою:

- Ми обчислюємо міру відстані під назвою WCSS (сума квадратів у кластері). Це говорить нам про те, наскільки розподілені точки даних у кожному кластері.
- Ми пробуємо різні значення k (кількість кластерів). Для кожного k ми запускаємо K-Means і обчислюємо WCSS.
- Ми будуємо графік з k на осі X і WCSS на осі Y.
- Визначення точки ліктя: коли ми збільшуємо k, WCSS зазвичай зменшується, оскільки ми створюємо більше кластерів, які мають тенденцію фіксувати більше варіацій даних. Однак настає момент, коли додавання додаткових кластерів призводить лише до незначного зниження WCSS. Тут ми спостерігаємо форму «ліктя» на графіку.

Перед elbow point: збільшення k значно зменшує WCSS, вказуючи на те, що нові кластери ефективно охоплюють більше мінливості даних.

Після elbow point: додавання додаткових кластерів призводить до мінімального зниження WCSS, що свідчить про те, що ці додаткові кластери можуть бути непотрібними.

Мета полягає в тому, щоб визначити точку, де швидкість зниження WCSS різко змінюється, вказуючи на те, що додавання більшої кількості кластерів (поза цією точкою) дає меншу віддачу. Ця точка «ліктя» вказує на оптимальну кількість кластерів.

Спотворення (Distortion)

Спотворення вимірює середню квадратну відстань між кожною точкою даних та центром кластера, до якого вона належить. Це показник того, наскільки добре кластери відображають дані. Менше значення спотворення вказує на кращу кластеризацію.

$Distortion = \frac{1}{n}\sum_{i=1}^{n} d(x_i, c_j)^2$, де:

- x_i – і-та точка даних,
- c_j – центр кластера, до якого належить
- $d(x_i, c_j)$ – евклідова відстань між точкою даних і її центром кластера.

1.5 Silhouette Analysis

Silhouette Analysis є одним із багатьох алгоритмів для визначення оптимальної кількості кластерів для алгоритму K-Means. В алгоритмі Silhouette ми припускаємо, що дані вже кластеризовано в k кластерів за допомогою техніки кластеризації. Потім для кожної точки даних ми визначаємо наступне:

C(i) - Кластер, призначений і-й точці даних

|C(i)| – Кількість точок даних у кластері, призначених і-й точці даних

a(i) – Показує, наскільки добре і-та точка даних присвоєна її кластеру (середня відстань між кожною точкою в кластері).

$$a(i) = \frac{1}{|C(i)|-1} \sum_{C(i),\, i \neq j} d(i,j)$$

b(i) – визначається як середня розбіжність з найближчим кластером, який не є даним кластером

$$b(i) = min_{i \neq j}\left(\frac{1}{|C(j)|} \sum_{j \in C(j)} d(i,j)\right)$$

На рис. 1 схематично зображено величини a(i) та b(i).

Silhouette Score s(i) визначається як $s(i) = \frac{b(i)-a(i)}{max(b(i),a(i))}$. Ми визначаємо середнє значення s(i) для кожного значення k і значення k, яке має максимальне значення s(i), вважається оптимальною кількістю кластерів для алгоритму.

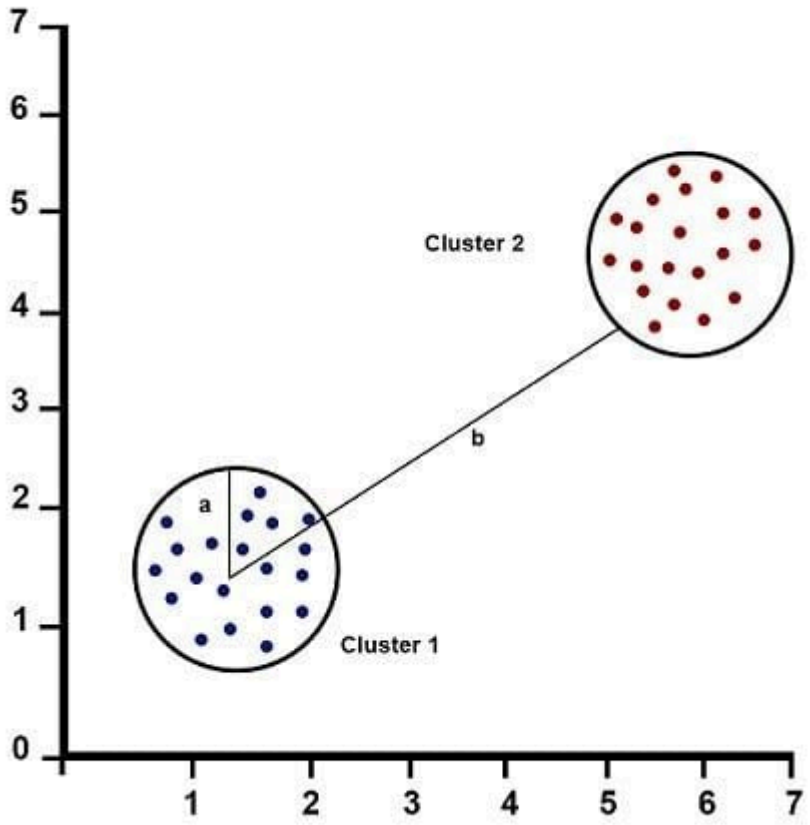

Рисунок 1 - величини Silhouette Score

Значення Silhouette Score коливається від -1 до 1. Нижче наведено інтерпретацію Silhouette Score.

1: Точки ідеально розподілені в кластері, і кластери легко розрізнити.

0: Кластери перекриваються.

-1: точки неправильно розподілені в кластері.

1.6 Principal Component Analysis

PCA — це статистичний метод, який був введений математиком Карлом Пірсоном у 1901 році. Він працює шляхом перетворення даних з високою вимірністю в простір з нижчою вимірністю, максимально зберігаючи дисперсію (або розкидання) даних у новому просторі. Це допомагає зберегти найбільш важливі закономірності та зв'язки в даних.

Примітка: він віддає перевагу напрямкам, де дані змінюються найбільше (адже більше змін = більше корисної інформації).

Кроки методу PCA:

1. Стандартизація даних

Необхідно, щоб усі ознаки мали однаковий масштаб, інакше змінні з більшими значеннями (наприклад, зарплата 0–100000) будуть домінувати над змінними з меншими значеннями (наприклад, вік 0–100).

Формула стандартизації:

$Z = \frac{X-\mu}{\sigma}$, де μ – середнє значення кожної ознаки, σ – стандартне відхилення кожної ознаки.

2. Обчислення коваріаційної матриці

Необхідно визначити, як ознаки змінюються разом. Це робиться за допомогою коваріаційної матриці, що показує взаємозв’язок між парами змінних.

Формула для обчислення коваріації між двома змінними

$$cov(x_1, x_2) = \frac{\sum_{i=1}^{n}(x_{1i}-\bar{x}_1)(x_{2i}-\bar{x}_2)}{n-1}$$

Результат може бути:

- Додатним: обидві змінні збільшуються разом.
- Від’ємним: одна змінна збільшується, а інша зменшується.
- Нульовим: відсутній лінійний взаємозв’язок.

3. Обчислення власних значень та власних векторів

PCA визначає нові осі (напрямки), уздовж яких дані мають найбільшу дисперсію.

Перша головна компонента (PC1) – напрямок найбільшого розкиду даних.

Друга головна компонента (PC2) – наступний найважливіший напрямок, перпендикулярний до PC1.

Для цього використовуються власні вектори та власні значення.

Якщо матриця A є квадратною, то власний вектор X та власне значення λ задовольняють рівняння $AX=\lambda X$.

Це означає, що:

- A лише масштабує X без зміни його напрямку.
- Власні вектори визначають "стабільні напрями" матриці A.

Рівняння можна записати у вигляді $(A-\lambda I)X=0$, де I – одинична матриця. Визначник цього рівняння має дорівнювати нулю: $|A-\lambda I|=0$
Це рівняння називається характеристичним рівнянням, а його розв’язання дає власні значення λ, після чого знаходяться відповідні власні вектори X.

4. Вибір головних компонент і перетворення даних

Зберігаються лише ті головні компоненти, які пояснюють найбільшу частину дисперсії (наприклад, 95%). Дані проектуються на ці компоненти, що дозволяє зменшити кількість вимірів без значної втрати інформації.

PCA є некерованим методом машинного навчання, оскільки не потребує міток цільових змінних. Його часто використовують для аналізу

даних, візуалізації та підвищення ефективності моделей машинного навчання.

1.6 Візуалізація роботи PCA

Припустимо, що є набір даних з двома ознаками: радіус (x-вісь) та площа (y-вісь), який зображено на рис. 2.

PCA знаходить напрямки найбільшої варіативності:

- PC_1 (перша головна компонента) – напрямок, уздовж якого дані мають найбільшу дисперсію.
- PC_2 (друга головна компонента) – напрямок, перпендикулярний до PC_1, який містить менше інформації.

Червоні штриховані лінії вказують на розкид даних уздовж різних напрямків. Оскільки розкид уздовж PC_1 більший, ніж уздовж PC_2, це означає, що PC_1 несе більше корисної інформації.

Дані (сині точки) проектуються на PC_1, що ефективно зменшує вимірність набору даних з двох (радіус, площа) до одного (PC_1).

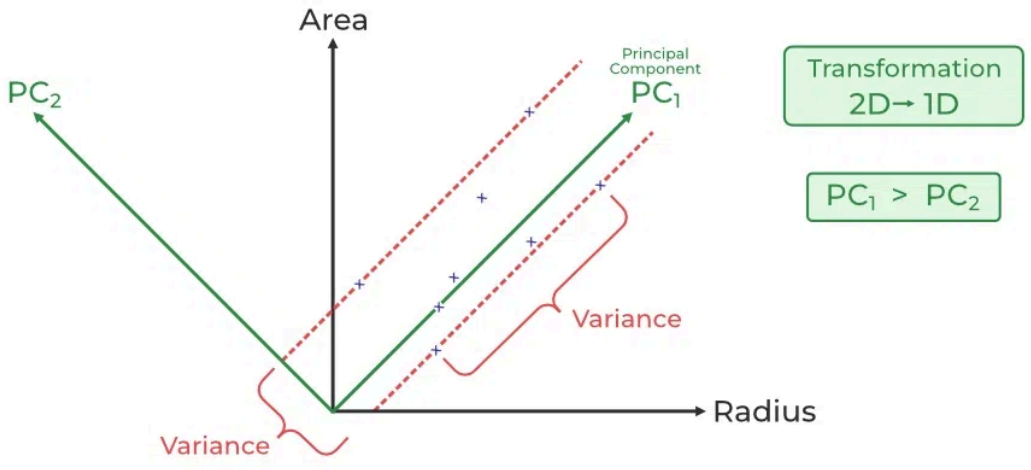

Рисунок 2 - 2D -> 1D трансформація за допомогою PCA

2. Реалізація

2.1 Data Preprocessing

Датасет поданий у форматі .csv з полями:

1. CustomerID - Unique ID assigned to the customer
2. Gender - Gender of the customer
3. Age - Age of the customer
4. Annual Income (k$) - Annual Income of the customee
5. Spending Score (1-100) - Score assigned by the mall based on customer behavior and spending nature

CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	31	17	40
6	Female	22	17	76
7	Female	35	18	6
8	Female	23	18	94
9	Male	64	19	3
10	Female	30	19	72
11	Male	67	19	14
12	Female	35	19	99
13	Female	58	20	15
14	Female	24	20	77

Рисунок 3 - вміст файлу Mall_Customers.csv

Використовуємо бібліотеку pandas для роботи з датасетом. Видаляємо ідентифікатор, оскільки він не впливає на кластеризацію. Вибираємо фічі для кластеризації: Age, Annual Income, Spending Score. Нормалізуємо дані за допомогою StandardScaler().

2.2 Exploratory Data Analysis (EDA)

Будуємо розподіли за Age, Annual Income, Spending Score, Gender; графіки для відображення зв’язків Annual Income - Spending score, Age -

Spending Score, Age - Annual Income; кореляційну матрицю; лінії тренду для розподілів. Згадані вище дані зображені на рис. 4 - рис. 10.

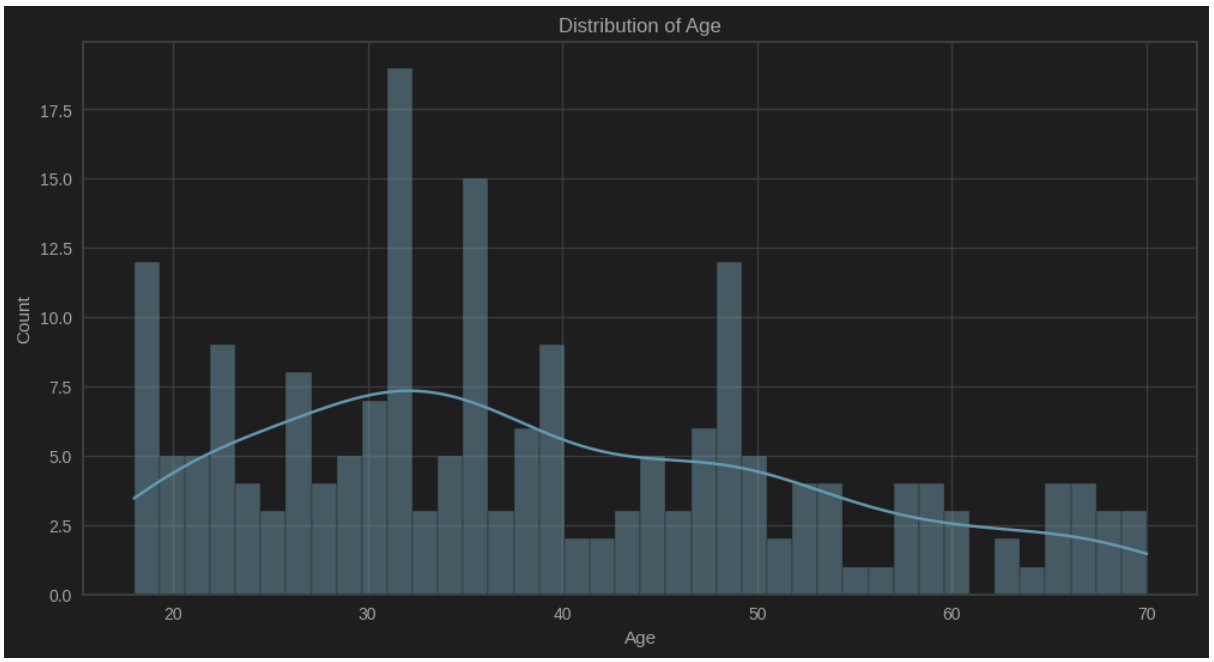

Рисунок 4 - Розподіл Age

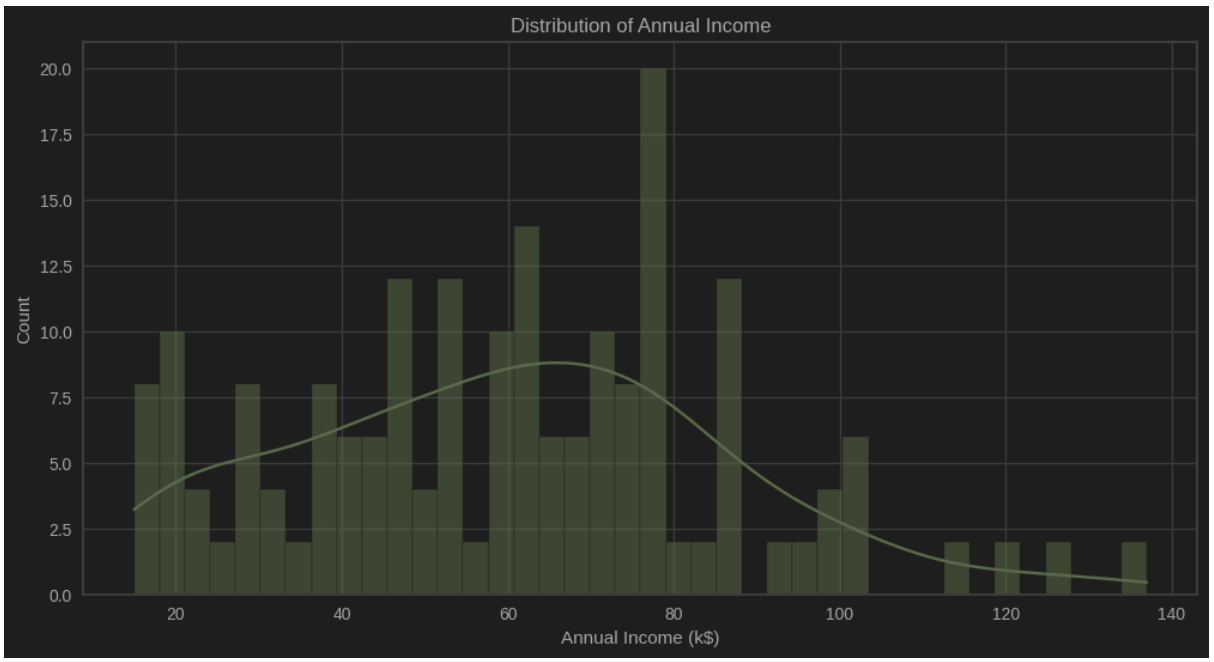

Рисунок 5 - Розподіл Annual Income

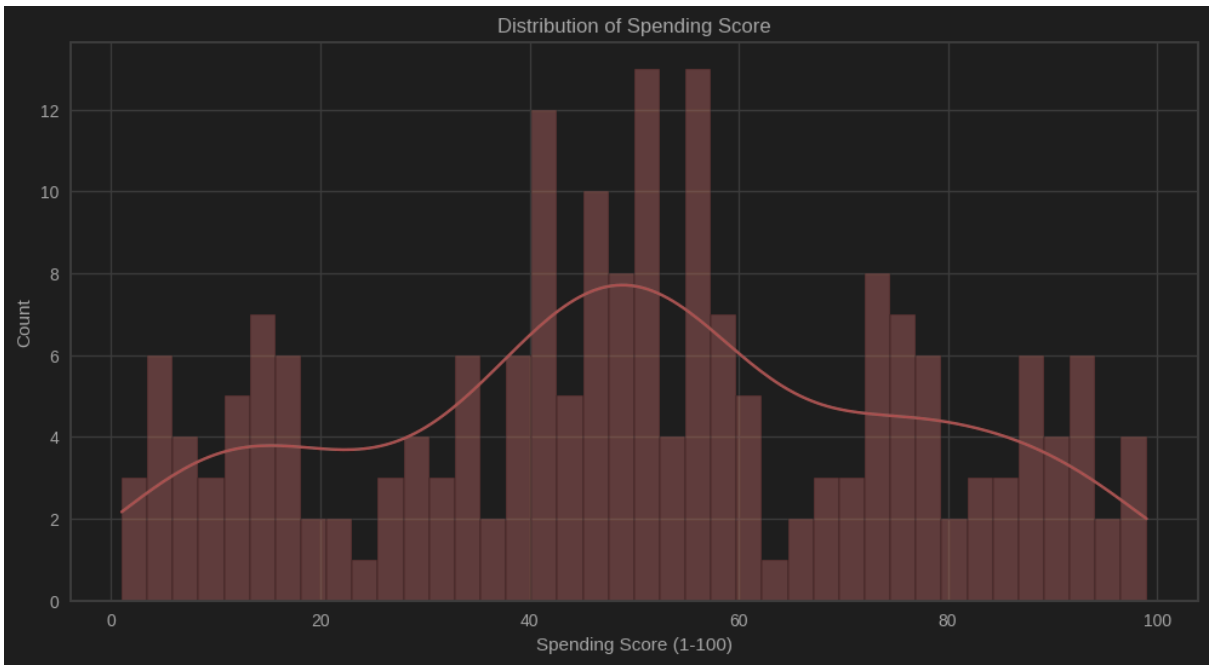

Рисунок 5 - Розподіл Spending Score

Рисунок 6 - Annual Income vs Spending Score

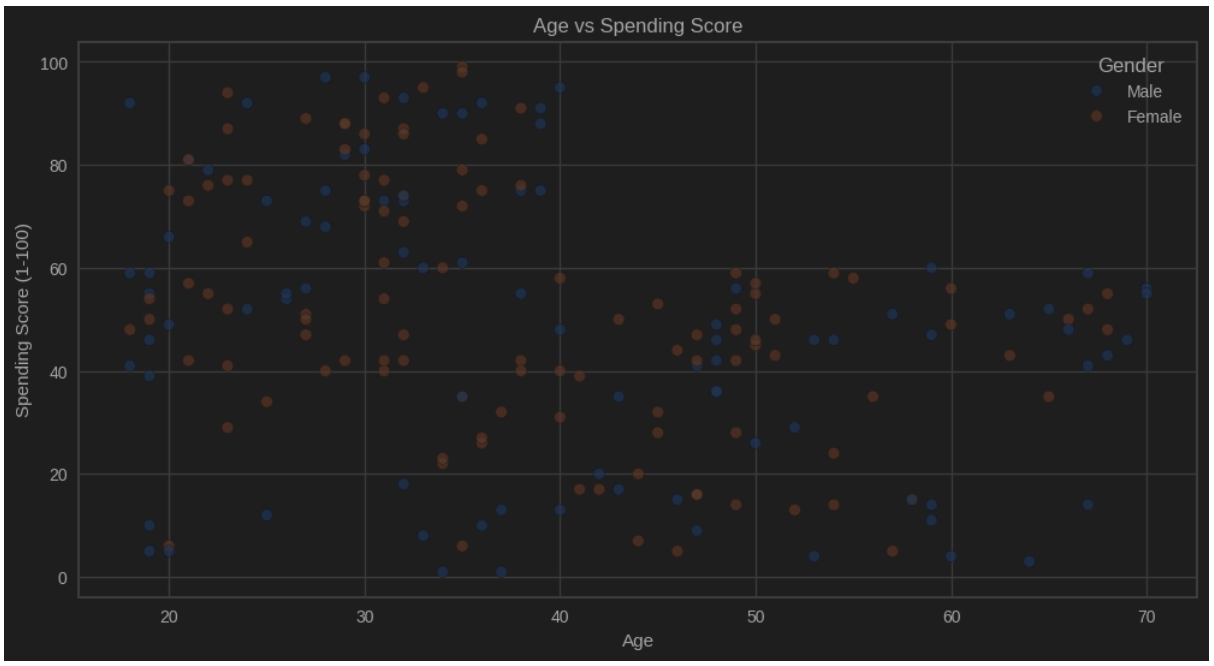

Рисунок 7 - Age vs Spending Score

Рисунок 8 - Age vs Annual Income

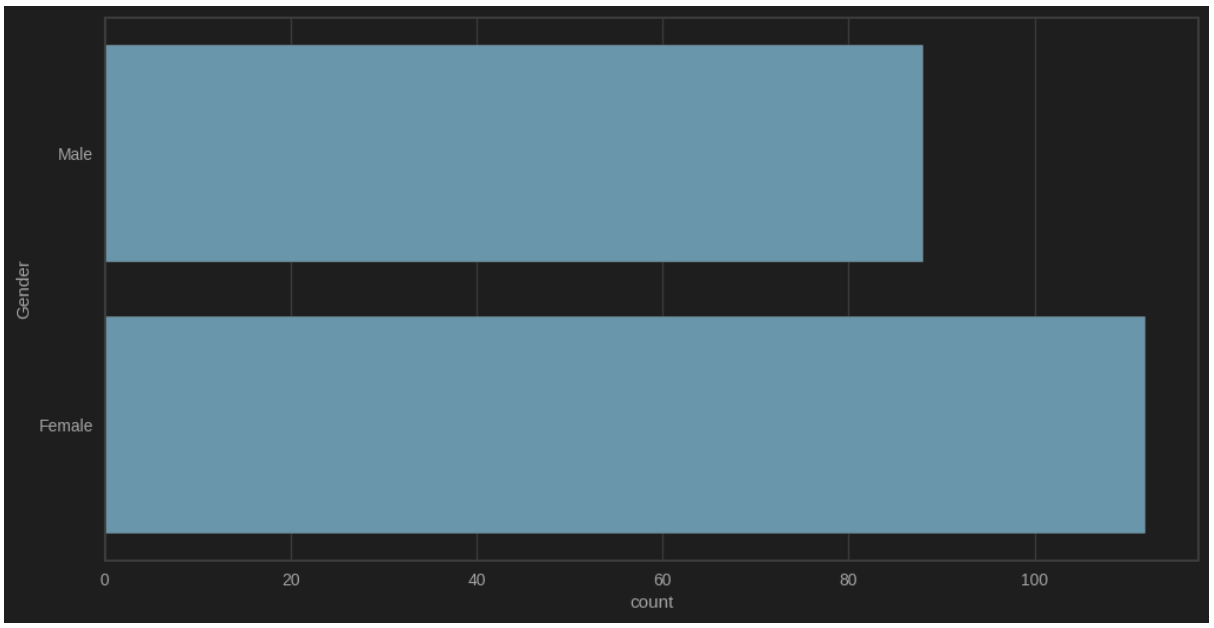

Рисунок 9 - Розподіл Gender

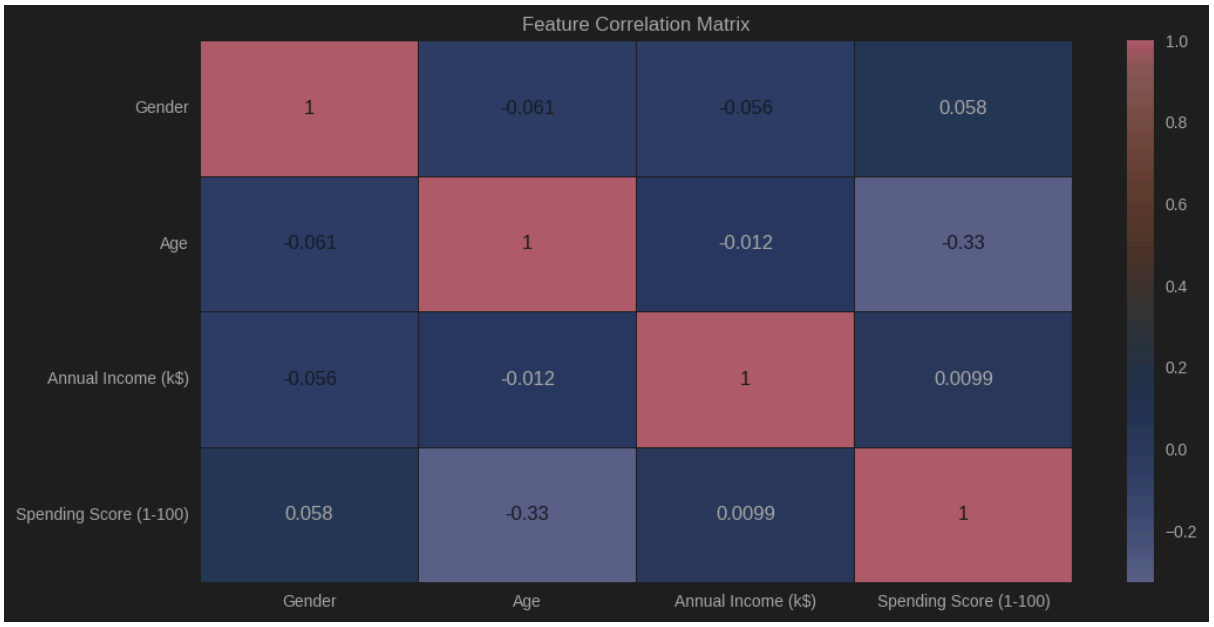

Рисунок 10 - Кореляційна матриця

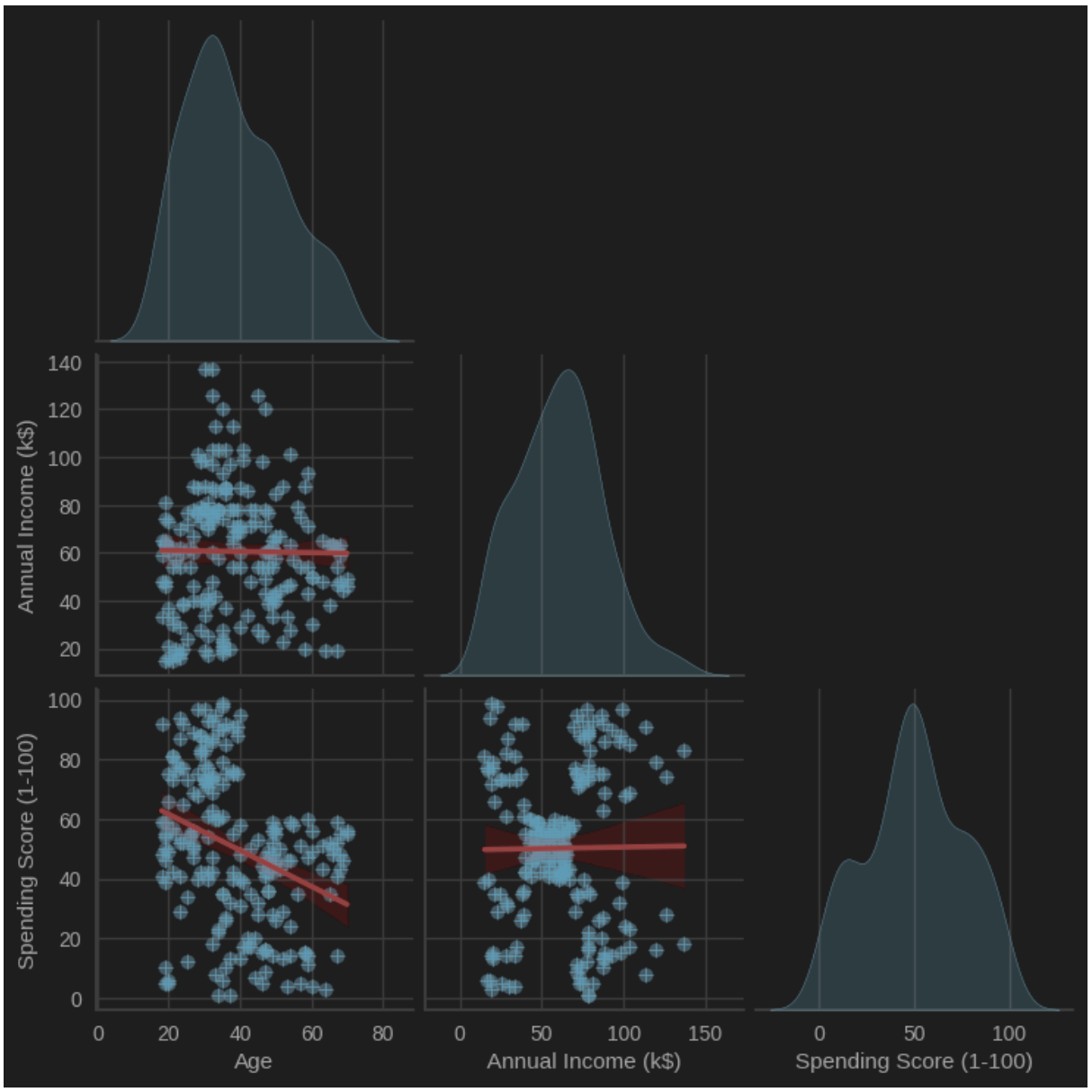

Рисунок 11 - Об’єднані графіки розподілів з лініями трендів

2.3 Implementing K-means Clustering

Використовуємо бібліотеку sklearn для імплементації алгоритму K-Means. Для визначення оптимального k будемо використовувати Elbow Method та Silhouette Analysis (бібліотека yellowbrick). Для кожного значення k рахуємо значення коефіцієнту спотворення та визначаємо точку “ліктя” на графіку, який зображений на рис. 12.

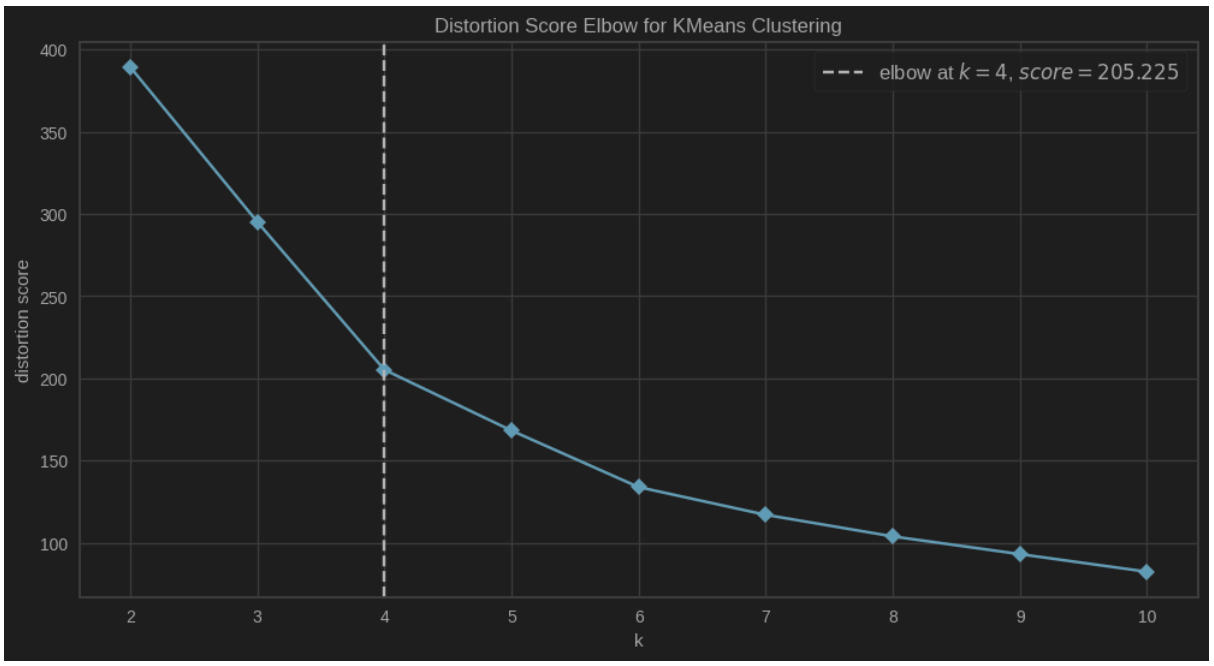

Рисунок 12 - Коефіцієнт спотворення для різник величин k

Бачимо, що за методом ліктя, оптимальне число кластерів - 4.

Аналогічно, рахуємо Silhouette Score для різних значень k. Графік Silhouette Score наведено на рис. 13.

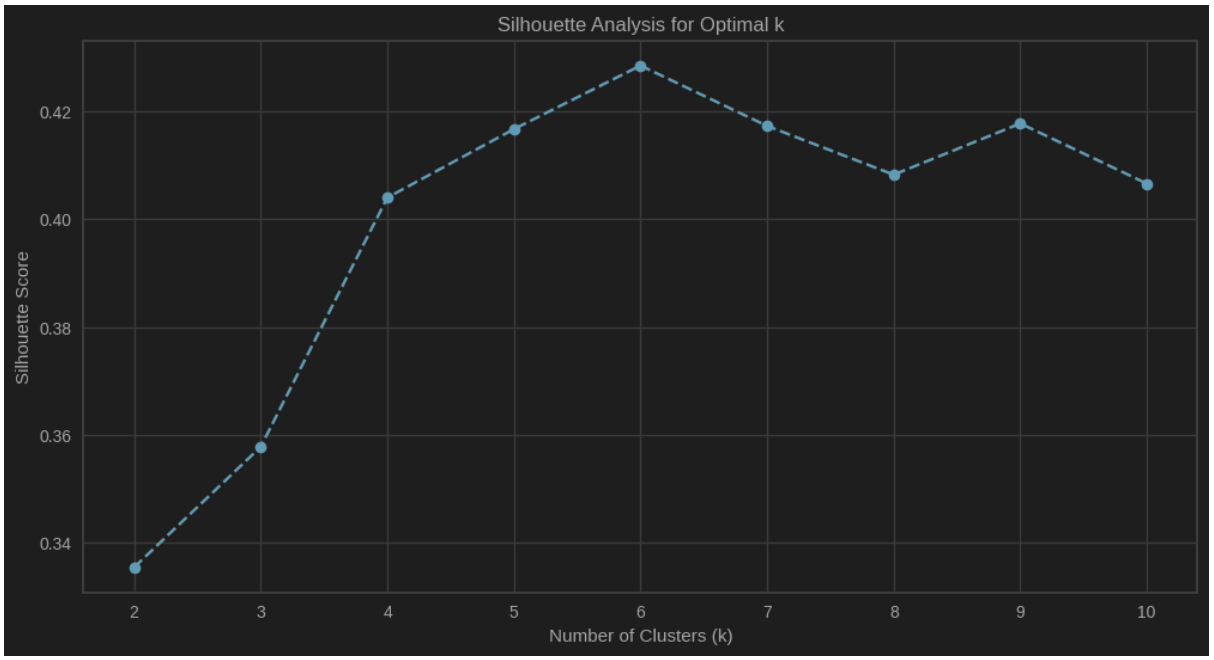

Рисунок 13 - Silhouette Score для різник величин k

Додатково проаналізуємо розміри та значення Silhouette Score для кожного кластеру для кожного k. Графіки наведені на рис. 14 - рис. 18.

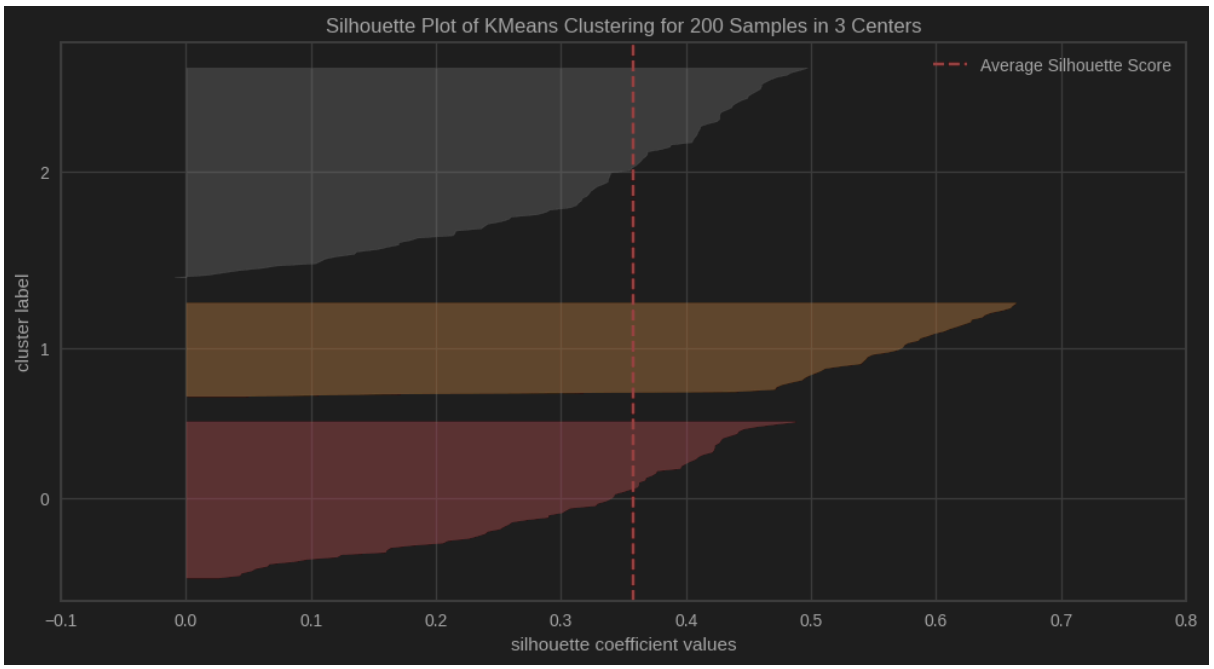

Рисунок 14 - Silhouette Score для k=3

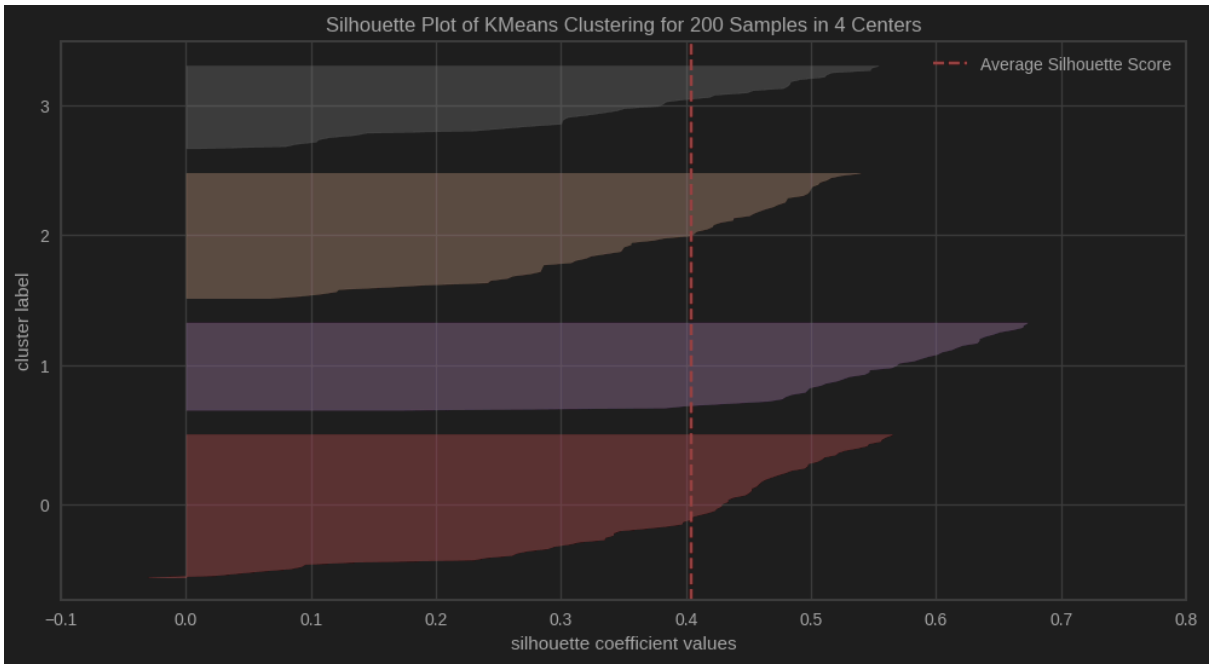

Рисунок 15 - Silhouette Score для k=4

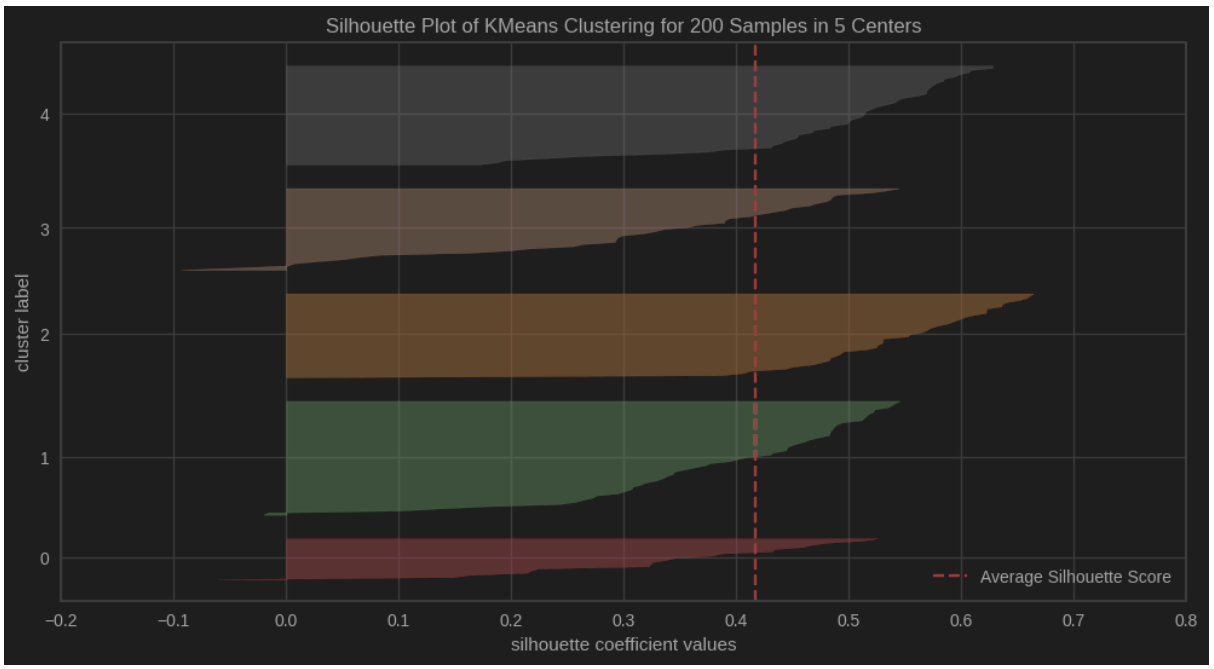

Рисунок 16 - Silhouette Score для k=5

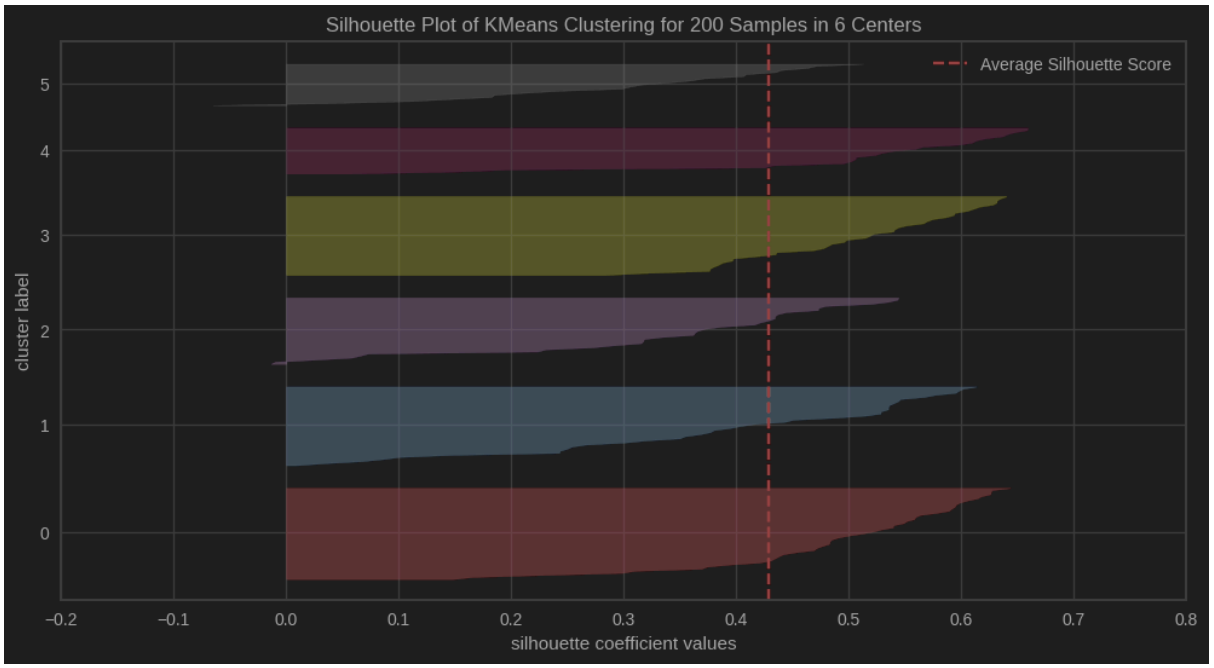

Рисунок 17 - Silhouette Score для k=6

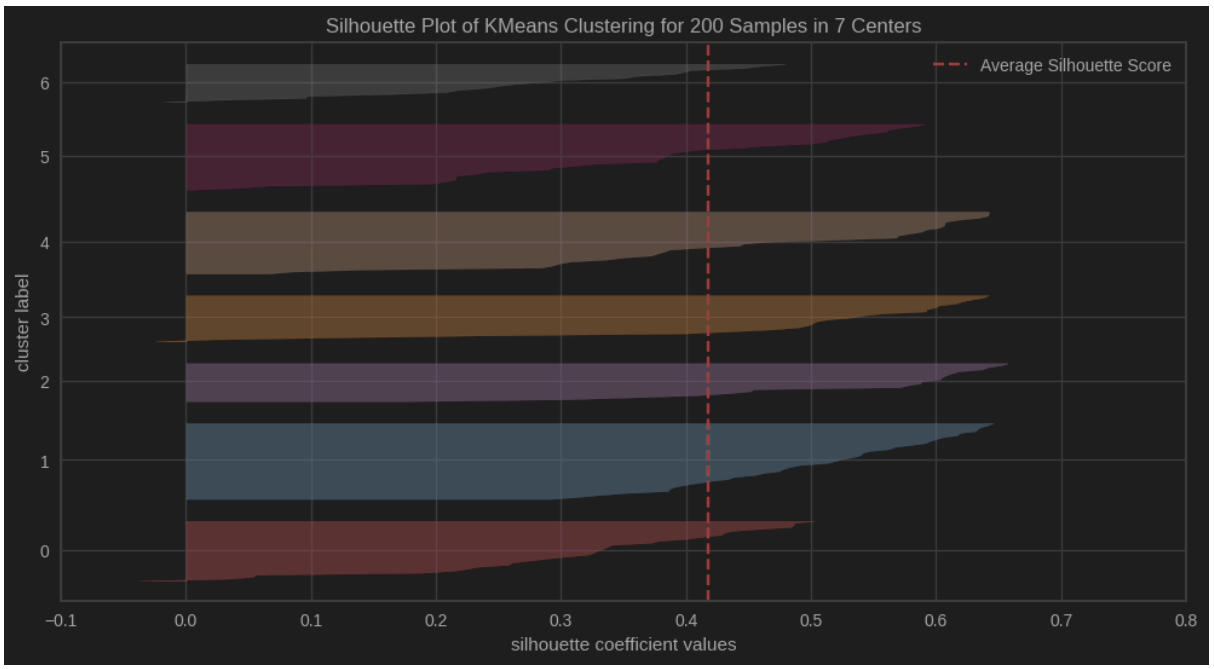

Рисунок 18 - Silhouette Score для k=7

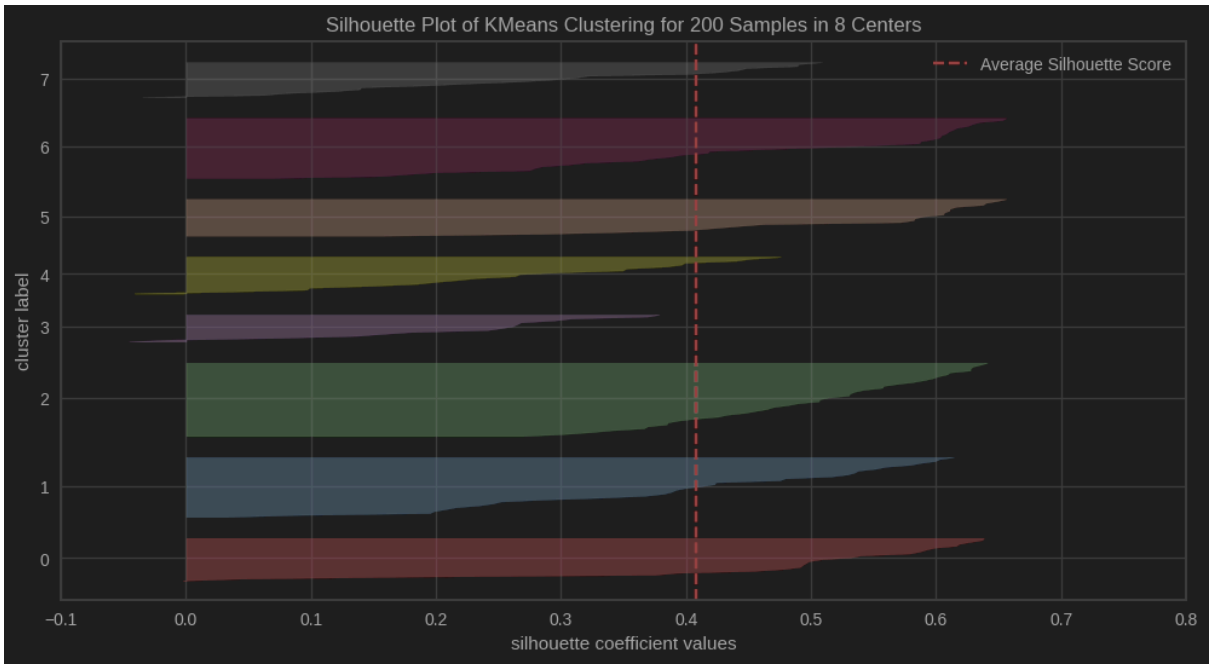

Рисунок 19 - Silhouette Score для k=8

Бачимо, що Silhouette Score починає зменшуватись після k=6. Для k=8 кожна точка у четвертому кластері має Silhouette Score менше, ніж середній для множини точок. Тому це значення k буде гірше, ніж інші. Для k=7 майже аналогічна ситуація відбувається з останнім кластером. Значення, які будуть нас задовольняти лежать в межах від 4 до 6. Покладемо k=4, так як точки в кластерах розподілені більш рівномірно,

ніж для значень k=5, k=6. Але залежно від потреб поставленої задачі може бути вибрана інша кількість кластерів.

2.4 Evaluation of Clusters

На рис. 20 наведена основна інформація про кластери.

```
Silhouette Score for k=4: 0.4040
         Gender      Age  Annual Income (k$)  Spending Score (1-100)  \
Cluster
0       0.56923 53.98462            47.70769                39.96923
1       0.55000 32.87500            86.10000                81.52500
2       0.59649 25.43860            40.00000                60.29825
3       0.50000 39.36842            86.50000                19.57895

         Silhouette Score
Cluster
0                 0.38189
1                 0.55097
2                 0.36759
3                 0.34151
Cluster Centroids:
       Age  Annual Income (k$)  Spending Score (1-100)  Cluster
0  1.08616            -0.49057                -0.39717        0
1 -0.42881             0.97485                 1.21609        1
2 -0.96249            -0.78476                 0.39203        2
3  0.03721             0.99011                -1.18876        3
```

Рисунок 20 - Інформація про кластери

Кластер 0 - найстарша аудиторія з середнім заробітком, які здійснюють помірні покупки.

Кластер 1 - молода аудиторія з високим заробітком, які витрачають багато. Потенційно може бути цільовою аудиторією для цілеспрямованого маркетингу.

Кластер 2 - наймолодша аудиторія з низьким заробітком, які витрачають більше, ніж в середньому всі покупці.

Кластер 3 - середня за віком аудиторія з високим заробітком, які витрачають найменше.

2.5 Visualization

На рис. 21 зображена кластеризація покупців в просторі розмірності 3.

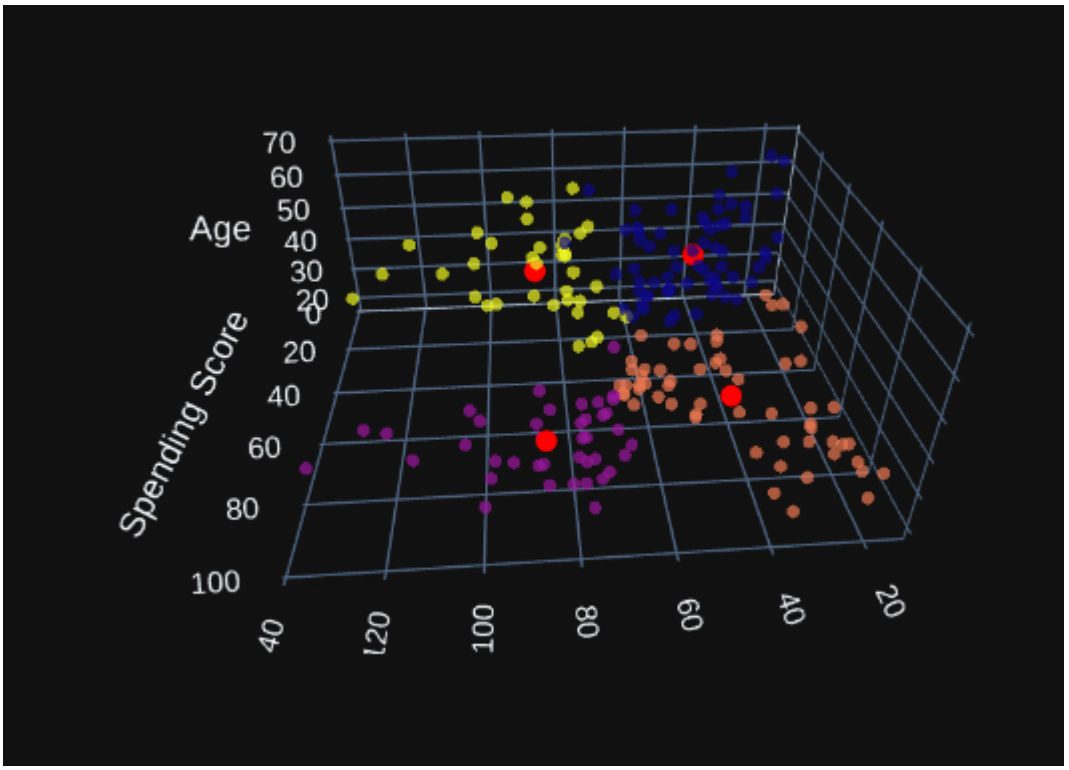

Рисунок 21 - Початкова кластеризація

Використаємо Principal Component Analysis для пониження розмірності до площини. У нас вийде дві компоненти, значення яких будуть лінійними комбінаціями початкових ознак. Коефіцієнти лінійних комбінацій зображено на рис. 22. Кластеризація в розмірності 2 зображена на рис. 23.

	Age	Annual Income (k$)	Spending Score (1-100)
PCA1	0.70638	-0.04802	-0.70620
PCA2	0.03014	0.99883	-0.03777

Рисунок 22 - Коефіцієнти лінійних комбінацій

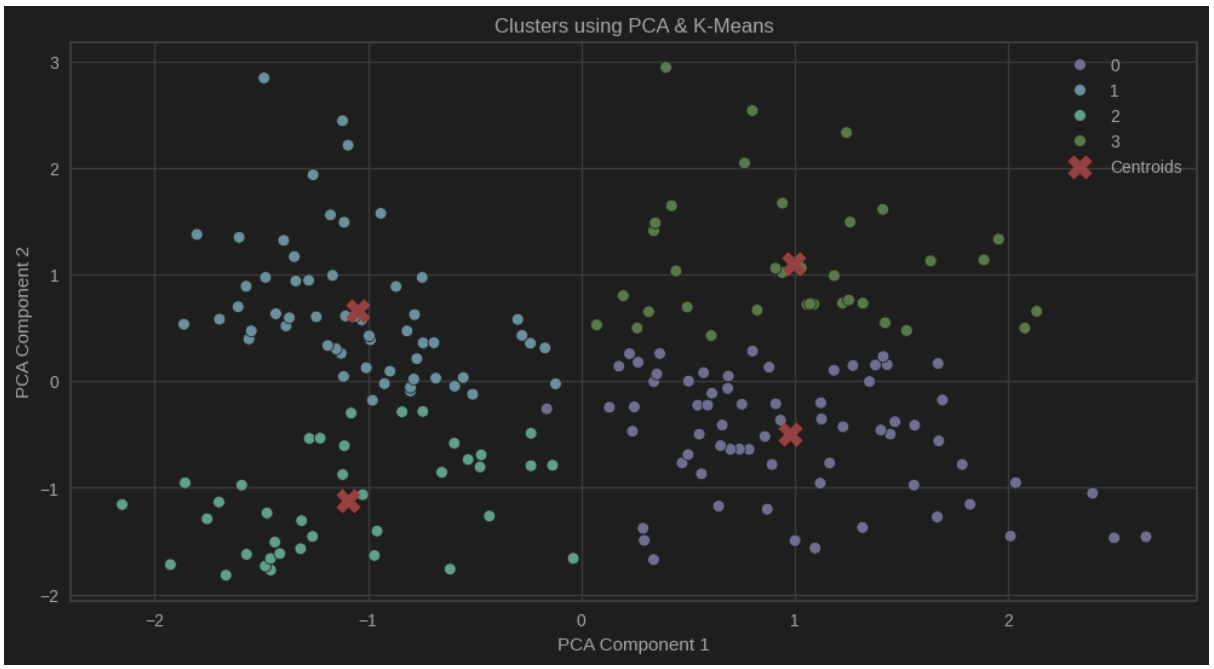

Рисунок 23 - Кластеризація після застосування PCA

Графіки для візуальної репрезентації кожного кластеру зображені на рис. 24 - рис. 29.

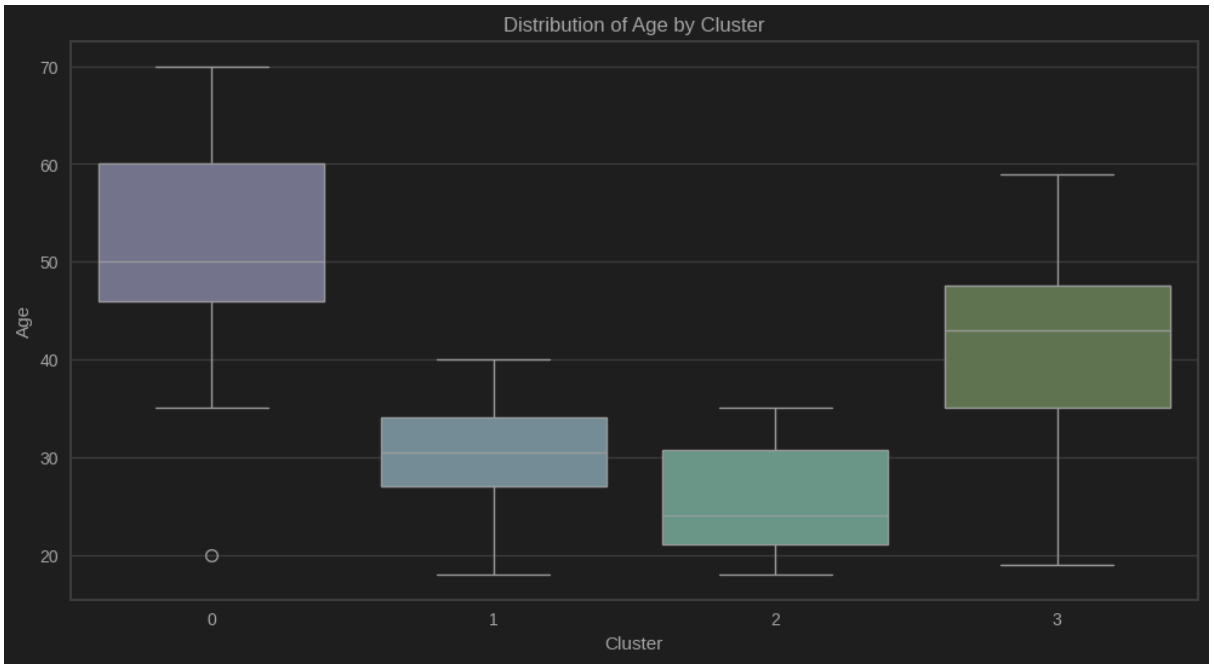

Рисунок 24 - Розподіл Age у кожному кластері

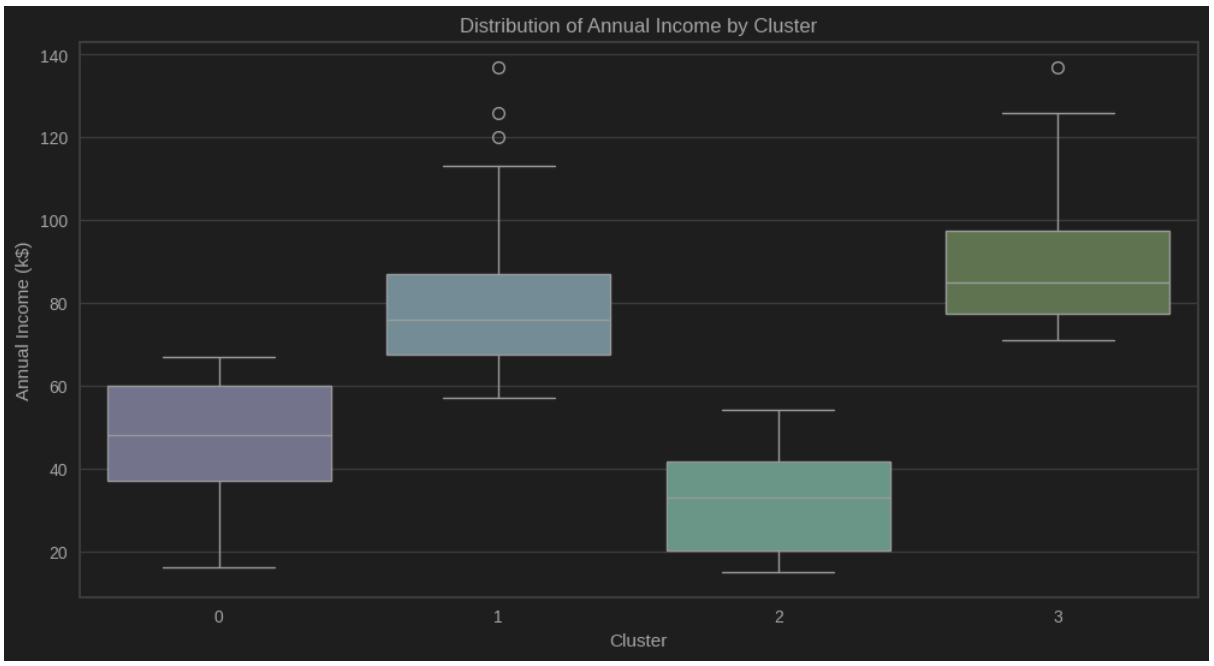

Рисунок 25 - Розподіл Annual Income у кожному кластері

Рисунок 26 - Розподіл Spending Score у кожному кластері

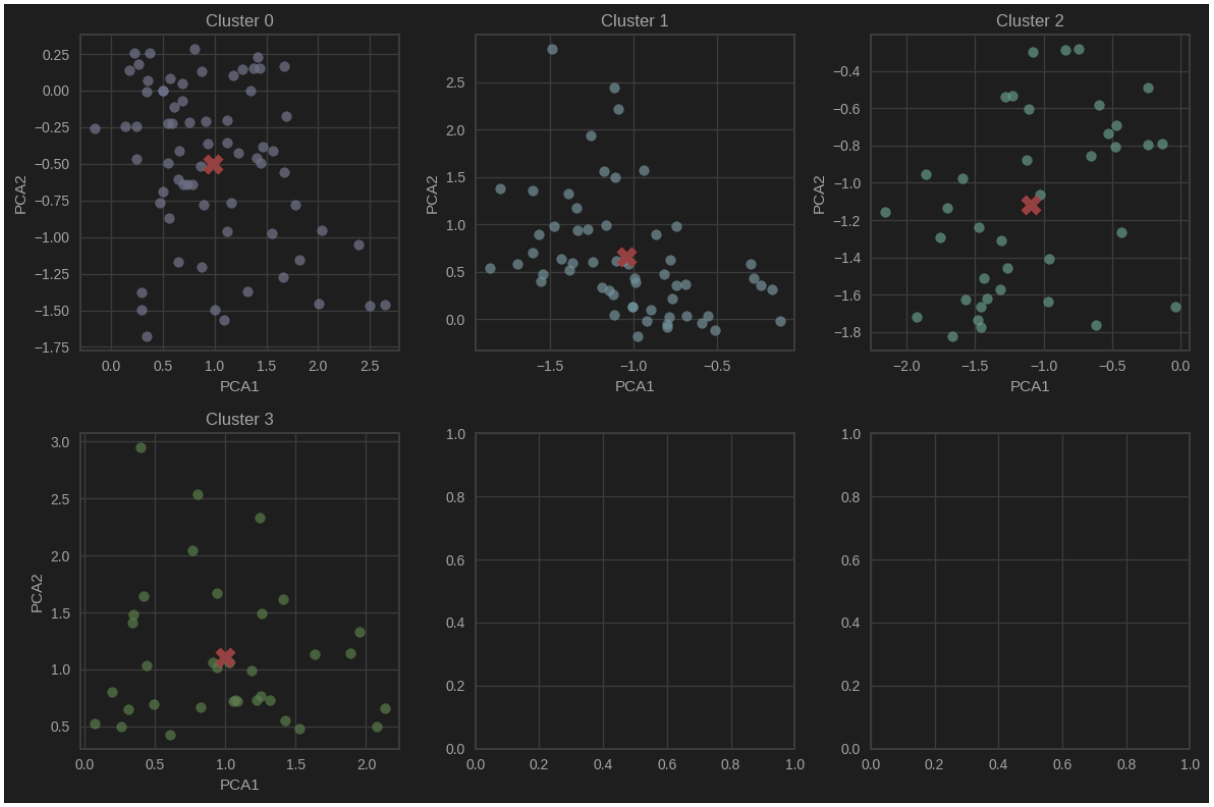

Рисунок 27 - Візуалізація всіх кластерів окремо

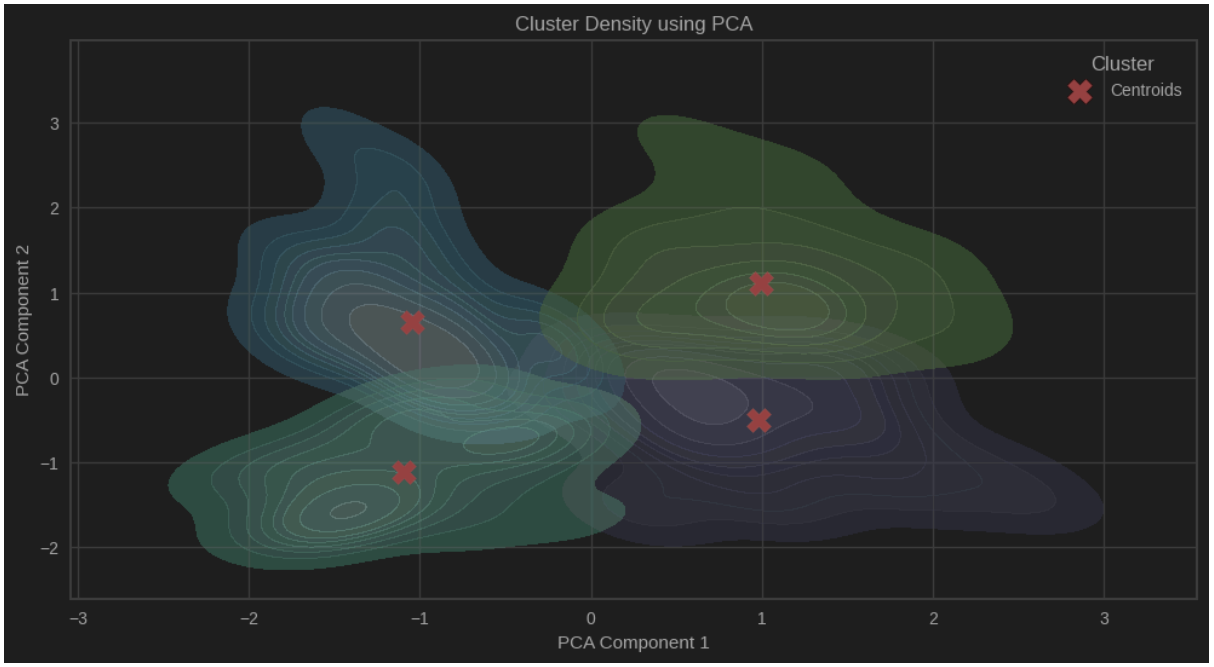

Рисунок 28 - Щільність кластерів

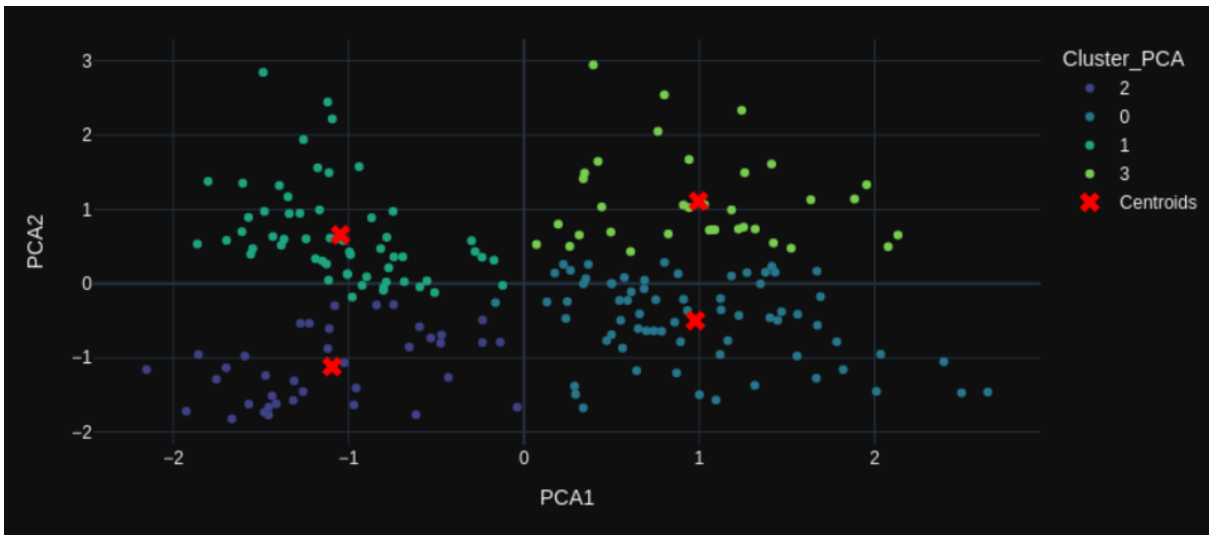

Рисунок 29 - Інтерактивна візуалізація кластерів

Висновки

В результаті виконання лабораторної роботи було розроблено функціональний додаток для сегментації клієнтів за допомогою алгоритму K-means. Додаток дозволяє групувати клієнтів на основі їхніх характеристик, що може бути використано для аналізу споживчої поведінки та розробки маркетингових стратегій.

Перспективою розвитку застосунку є інтеграція альтернативних методів кластеризації, таких як DBSCAN та агломеративна кластеризація, а також використання глибшого аналізу ознак, зокрема з урахуванням демографічних даних та поведінкових патернів.

Для виконання кластеризації було використано алгоритм K-means, а також реалізовано такі завдання:

- Проведено попередню обробку та нормалізацію даних.
- Виконано дослідницький аналіз та візуалізацію розподілу ключових ознак.
- Визначено оптимальну кількість кластерів за допомогою методу ліктя та силуетного аналізу.
- Застосовано алгоритм K-means для сегментації клієнтів.
- Оцінено якість кластеризації за метриками та проаналізовано характеристики кожного кластеру.
- Побудовано візуалізації результатів, включаючи зменшення розмірності за допомогою PCA.

Розроблений додаток може бути використаний для персоналізації рекламних кампаній, покращення обслуговування клієнтів та виявлення нових бізнес-можливостей.

Додатки

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score, silhouette_samples
from sklearn.decomposition import PCA
from yellowbrick.cluster import KElbowVisualizer, SilhouetteVisualizer
import plotly.express as px

df = pd.read_csv("~/Downloads/Mall_Customers.csv")
df = df.drop(columns=['CustomerID'])
#print(df.isnull().sum())

# Features selections
features = ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']
X = df[features]

# Data Scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Distributions of features
plt.figure(figsize=(12, 6))
sns.histplot(df['Age'], bins=40, kde=True)
plt.title('Distribution of Age')
plt.show()

plt.figure(figsize=(12, 6))
sns.histplot(df['Annual Income (k$)'], bins=40, kde=True, color='g')
plt.title('Distribution of Annual Income')
plt.show()

plt.figure(figsize=(12, 6))
sns.histplot(df['Spending Score (1-100)'], bins=40, kde=True, color='r')
plt.title('Distribution of Spending Score')
```

```
plt.show()

# Relations between features
plt.figure(figsize=(12, 6))
sns.scatterplot(x=df['Annual Income (k$)'], y=df['Spending Score (1-100)'],
hue=df['Gender'], palette='coolwarm', alpha=0.7)
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.title('Annual Income vs Spending Score')
plt.legend(title='Gender')
plt.show()

plt.figure(figsize=(12, 6))
scatter = sns.scatterplot(x=df['Age'], y=df['Spending Score (1-100)'],
hue=df['Gender'], palette='coolwarm', alpha=0.7)
plt.xlabel('Age')
plt.ylabel('Spending Score (1-100)')
plt.title('Age vs Spending Score')
plt.legend(title='Gender')
plt.show()

plt.figure(figsize=(12, 6))
sns.scatterplot(x=df['Age'], y=df['Annual Income (k$)'], hue=df['Gender'],
palette='coolwarm', alpha=0.7)

plt.xlabel('Age')
plt.ylabel('Annual Income (k$)')
plt.title('Age vs Annual Income')
plt.legend(title='Gender')
plt.show()

plt.figure(1 , figsize = (12, 6))
sns.countplot(y = 'Gender' , data = df)
plt.show()

df['Gender'] = df['Gender'].map({'Male': 0, 'Female': 1})
```

```
#Correlation matrix
plt.figure(figsize=(12, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Feature Correlation Matrix')
plt.show()

# Pairplot with trend lines
df2 = df.copy()
df2 = df2.drop(columns=["Gender"])
plt.figure(figsize=(12, 6))
g = sns.pairplot(df2, diag_kind='kde', markers='+', corner=True)
g.map_lower(sns.regplot, scatter_kws={'alpha':0.5}, line_kws={'color':'red'})
plt.show()

# Elbow method with yellowbrick
plt.figure(figsize=(12, 6))
model = KMeans(random_state=42, n_init=10)
visualizer = KElbowVisualizer(model, k=(2, 11), metric='distortion',
timings=False)
visualizer.fit(X_scaled)
visualizer.show()

# Silhouette Analysis
silhouette_scores = []
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    labels = kmeans.fit_predict(X_scaled)
    silhouette_scores.append(silhouette_score(X_scaled, labels))

plt.figure(figsize=(12, 6))
plt.plot(range(2, 11), silhouette_scores, marker='o', linestyle='--')
plt.xlabel('Number of Clusters (k)')
plt.ylabel('Silhouette Score')
plt.title('Silhouette Analysis for Optimal k')
plt.show()

# Silhouette score visualisation with yellowbrick
```

```
for optimal_k in range(3, 9):
    plt.figure(figsize=(12, 6))
    kmeans = KMeans(n_clusters=optimal_k, random_state=42, n_init=10)
    visualizer = SilhouetteVisualizer(kmeans)
    visualizer.fit(X_scaled)
    visualizer.show()

# K-Means algorithm
optimal_k = 4
kmeans = KMeans(n_clusters=optimal_k, random_state=42, n_init=10)
df['Cluster'] = kmeans.fit_predict(X_scaled)

# Silhouette score for chosen k
silhouette_avg = silhouette_score(X_scaled, df['Cluster'])
print(f'Silhouette Score for k={optimal_k}: {silhouette_avg:.4f}')

# Silhouette score for each cluster
silhouette_values = silhouette_samples(X_scaled, df['Cluster'])
df['Silhouette Score'] = silhouette_values

# Cluster analysis
centroids = pd.DataFrame(kmeans.cluster_centers_, columns=features)
print(df.groupby('Cluster').mean())
centroids['Cluster'] = range(optimal_k)
print("Cluster Centroids:")
print(centroids)

# 3D visualization of clusters
centroids_original = scaler.inverse_transform(kmeans.cluster_centers_)

df_clustered = df.copy()
df_clustered['Cluster'] = df['Cluster']

fig = px.scatter_3d(df_clustered,
                    x='Annual Income (k$)',
                    y='Spending Score (1-100)',
                    z='Age',
```

```
                color='Cluster',
                title='3D Visualization of Clusters with Centroids',
                labels={'Annual Income (k$)': 'Annual Income',
                    'Spending Score (1-100)': 'Spending Score',
                    'Age': 'Age'},
                opacity=0.7)
fig.update_traces(marker=dict(size=3))
fig.add_scatter3d(x=centroids_original[:, 1],
                y=centroids_original[:, 2],
                z=centroids_original[:, 0],
                mode='markers+text',
                marker=dict(size=5, color='red'),
                name='Centroids')
fig.show()
palette = sns.color_palette("viridis", optimal_k)

# Using PCA to visualize in 2d
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
clusters_pca = kmeans.fit_predict(X_pca)

df['PCA1'] = X_pca[:, 0]
df['PCA2'] = X_pca[:, 1]
df['Cluster_PCA'] = clusters_pca

# Clusters visualization in 2d
plt.figure(figsize=(12, 6))
sns.scatterplot(x=df['PCA1'], y=df['PCA2'], hue=df['Cluster_PCA'],
palette=palette, s=50)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
        c='red', marker='X', s=200, label='Centroids')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.title('Clusters using PCA & K-Means')
plt.legend()
plt.show()
```

```
print(pd.DataFrame(pca.components_, columns=features, index=['PCA1',
'PCA2']))

# Visualization of features distribution for each cluster
plt.figure(figsize=(12, 6))
sns.boxplot(x=df['Cluster_PCA'], y=df['Age'], palette=palette)
plt.title('Distribution of Age by Cluster')
plt.xlabel('Cluster')
plt.ylabel('Age')
plt.show()

plt.figure(figsize=(12, 6))
sns.boxplot(x=df['Cluster_PCA'], y=df['Annual Income (k$)'], palette=palette)
plt.title('Distribution of Annual Income by Cluster')
plt.xlabel('Cluster')
plt.ylabel('Annual Income (k$)')
plt.show()

plt.figure(figsize=(12, 6))
sns.boxplot(x=df['Cluster_PCA'], y=df['Spending Score (1-100)'],
palette=palette)
plt.title('Distribution of Spending Score by Cluster')
plt.xlabel('Cluster')
plt.ylabel('Spending Score (1-100)')
plt.show()

# Another clusters visualization
fig, axes = plt.subplots(2, 3, figsize=(12, 8))
axes = axes.flatten()

for cluster in range(optimal_k):
    ax = axes[cluster]
    subset = df[df['Cluster_PCA'] == cluster]
    ax.scatter(subset['PCA1'], subset['PCA2'], alpha=0.7, color=palette[cluster])
    ax.scatter(kmeans.cluster_centers_[cluster, 0],
kmeans.cluster_centers_[cluster, 1],
               color='red', marker='X', s=200, label='Centroid')
```

```
    ax.set_title(f"Cluster {cluster}")
    ax.set_xlabel("PCA1")
    ax.set_ylabel("PCA2")

plt.tight_layout()
plt.show()

# Clusters density visualization
plt.figure(figsize=(12, 6))
sns.kdeplot(data=df, x="PCA1", y="PCA2", hue="Cluster_PCA", fill=True,
alpha=0.5, palette=palette)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
            c='red', marker='X', s=200, label='Centroids')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.title('Cluster Density using PCA')
plt.legend(title='Cluster')
plt.show()
df["Cluster_PCA"] = df["Cluster_PCA"].astype(str)

# Interactive clusters visualization
fig = px.scatter(df, x="PCA1", y="PCA2", color="Cluster_PCA",
hover_data=["Age", "Annual Income (k$)", "Spending Score (1-100)"],
                 title="PCA Clusters Visualization",
color_discrete_sequence=palette.as_hex())
fig.add_scatter(x=kmeans.cluster_centers_[:, 0], y=kmeans.cluster_centers_[:,
1], mode='markers',
                marker=dict(symbol='x', color='red', size=12), name='Centroids')
fig.show()
```

Джерела

1. Principal Component Analysis (PCA) [Електронний ресурс] - Режим доступу до ресурсу: https://www.geeksforgeeks.org/principal-component-analysis-pca/
2. Elbow Method in K-Means Clustering [Електронний ресурс] - Режим доступу до ресурсу: https://builtin.com/data-science/elbow-method
3. A Tutorial on Principal Component Analysis [Електронний ресурс] - Режим доступу до ресурсу: https://user.eng.umd.edu/~jzsimon/biol708L/ref/ShlensPCATutorial.pdf
4. Silhouette Score [Електронний ресурс] - Режим доступу до ресурсу: https://how.dev/answers/what-is-silhouette-score
5. Elbow Method for optimal value of k in KMeans [Електронний ресурс] - Режим доступу до ресурсу: https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/
6. MacQueen (1967). Some methods for classification and analysis of multivariate observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press: 281—297.
7. Silhouette Algorithm to determine the optimal value of k [Електронний ресурс] - Режим доступу до ресурсу: https://www.geeksforgeeks.org/silhouette-algorithm-to-determine-the-optimal-value-of-k/
8. Sklearn library [Електронний ресурс] - Режим доступу до ресурсу: https://scikit-learn.org/stable/