КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп’ютерних наук та кібернетики

Звіт про виконання
Лабораторної роботи на тему
“ТЕСТУВАННЯ, ДОСЛІДЖЕННЯ ТА АНАЛІЗ ОСНОВНИХ АЛГОРИТМІВ КЛАСТЕРИЗАЦІЇ НАБОРІВ ЧИСЛОВИХ ДАНИХ”

Підготував:

студент групи ММШІ-1

Волохович Ігор

https://github.com/antomys/ClusterizingData

Київ 2022

ЗМІСТ

Постановка задачі

1. Обрати для аналізу 3-4 достатньо різні алгоритми кластеризації (за бажанням виконавця число алгоритмів можна збільшити). Узгодити вибір алгоритмів з колегами так, щоб у різних виконавців було не більше 1-2 спільних тем.

2. Порівняти результати роботи цих алгоритмів на різних наборах даних.

Набори даних формуються виконавцем:

– детерміновано

– стохастично (із застосуванням різних датчиків випадкових чисел)

– за можливістю – використати реальні набори даних.

Набори даних розподіляються на різні типи залежно від форми майбутніх кластерів.

Способи формування наборів даних або типів наборів даних можуть бути, наприклад, такими.

Кожному кластеру відповідає:

1) коло (куля, багатовимірна куля) певного радіусу, яке містить усі точки цього кластера;

2) еліпс (багатовимірний еліпс);

– осі всіх еліпсів взаємно паралельні; – —"—"—" непаралельні.

3) квадрат (куб, багатовимірний куб)

– сторони всіх квадратів взаємно паралельні; – —"—"—" непаралельні.

4) прямокутник (паралелепіпед, багатовимірний паралелепіпед)

– сторони всіх прямокутників взаємно паралельні;

– —"—"—" непаралельні.

5) різні несиметричні утворення (сукупності точок), наприклад, подібні до літер Г, С, П, Т, Е, Х та ін.

6) різні суміші попередніх типів;

7) інші варіанти (на вибір виконавця), можливість виявити свою фантазію та дослідницьку

Здібність.

3. Дослідити обрані алгоритми на "неперервність" (або стабільність):

Реалізувавши певний алгоритм А для певного обраного набору даних Д, зробити δ-зміну цього набору Д, перемістивши одну з точок набору Д на "δ-невелику" відстань.

Застосувати повторно алгоритм А для δ-модифікованого набору Д'. З'ясувати, чи зміняться результати кластеризації, і якщо зміняться, то наскільки істотно.

Повторити цю процедуру для різних точок (наборів точок) і різних значень δ.

4. На всіх етапах застосовувати різні методи візуалізації початкового набору даних, проміжних та остаточних результатів кластеризації.

5. Визначати та порівнювати час та необхідний об'єм пам'яті для кожної реалізації.

6. Написати звіт, який повинен містити:

- детальні змістовні описи обраних для аналізу алгоритмів;
- таблиці і діаграми, в яких зібрано результати дослідження;
- висновки та рекомендації щодо застосування досліджених алгоритмів;
- список джерел, використаних для виконання роботи.

Детальні змістовні описи обраних для аналізу алгоритмів

В даному розділі будуть розглянуті та описані обрані для кластеризації алгоритми.

Кластеризація - це поділ безлічі вхідних векторів на групи (кластери) за рівнем схожості один на одного.

Кластеризація набуває цінності тоді, коли вона виступає одним з етапів аналізу даних, побудови закінченого аналітичного рішення. Аналітику часто легше виділити групи подібних об'єктів, вивчити їх особливості і побудувати кожної групи окрему модель, ніж створювати одну загальну модель всім даних. Таким прийомом постійно користуються у маркетингу, виділяючи групи клієнтів, покупців, товарів хороших і розробляючи кожної їх окрему стратегію.

Заходи відстаней

Для того, щоб порівнювати два об'єкти, необхідно мати критерій , на підставі якого відбуватиметься порівняння. Зазвичай, таким критерієм є відстань між об'єктами.

Є безліч заходів відстані, розглянемо кілька з них:

1. Відстань Евклідова — найбільш поширена відстань. Воно є геометричною відстанню у багатовимірному просторі.
2. Квадрат евклідової відстані. Іноді може виникнути бажання звести в квадрат стандартну евклідову відстань, щоб надати більших ваг більш віддаленим один від одного об'єктам.

3. Відстань міських кварталів (манхеттенська відстань).Ця відстань є просто середнім різницями по координатам. Здебільшого цей захід відстані призводить до таких же результатів, як і для звичайної відстані Евкліда. Однак зазначимо, що для цього заходу вплив окремих великих різниць (викидів) зменшується (оскільки вони не зводяться у квадрат).
4. Відстань Чебишева. Ця відстань може виявитись корисною, коли бажають визначити два об'єкти як «різні», якщо вони розрізняються за якоюсь однією координатою (якимось виміром).
5. Ступінна відстань. Іноді бажають прогресивно збільшити або зменшити вагу, що належить до розмірності, на яку відповідні об'єкти сильно відрізняються. Це може бути досягнуто з використанням статичної відстані.

Вибір відстані (критерію схожості) лежить повністю на дослідникові. При виборі різних заходів результати кластеризації можуть суттєво відрізнятись.

Алгоритм k-means (k-середніх)

Найбільш простий, але водночас досить неточний метод кластеризації у класичній реалізації. Він розбиває безліч елементів векторного простору на відоме число кластерів k. Дія алгоритму така, що він прагне мінімізувати середньоквадратичне відхилення на точках кожного кластера. Основна ідея полягає в тому, що на кожній ітерації перераховується центр мас для кожного кластера, отриманого на попередньому кроці, потім вектори розбиваються на кластери знову відповідно до того, який з нових центрів виявився ближчим за обраною метрикою. Алгоритм завершується, коли на якійсь ітерації немає зміни кластерів.

Проблеми алгоритму k-means:

- Необхідно заздалегідь знати кількість кластерів.Мною було запропоновано метод визначення кількості кластерів, що грунтувався на знаходженні кластерів, розподілених за якимось законом (у моєму випадку все зводилося до нормального закону). Після цього виконувався класичний алгоритм k-means, який давав точніші результати.
- **Алгоритм дуже чутливий до вибору початкових центрів кластерів.** Класичний варіант має на увазі випадковий вибір класторів, що дуже часто було джерелом похибки. Як варіант рішення необхідно проводити дослідження об'єкта для більш точного визначення центрів початкових кластерів. У моєму випадку на початковому етапі пропонується приймати як центи найвіддаленіші точки кластерів.
- Не справляється із завданням, коли об'єкт належить до різних кластерів рівною мірою або не належить жодному.

Кластериза́ція ме́тодом k-сере́дніх (англ. k-means clustering) — популярний метод кластеризації, — впорядкування множини об'єктів в порівняно однорідні групи. Винайдений в 1950-х роках математиком Гуґо Штайнгаузом і майже одночасно Стюартом Ллойдом. Особливу популярність отримав після виходу роботи МакКвіна.

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

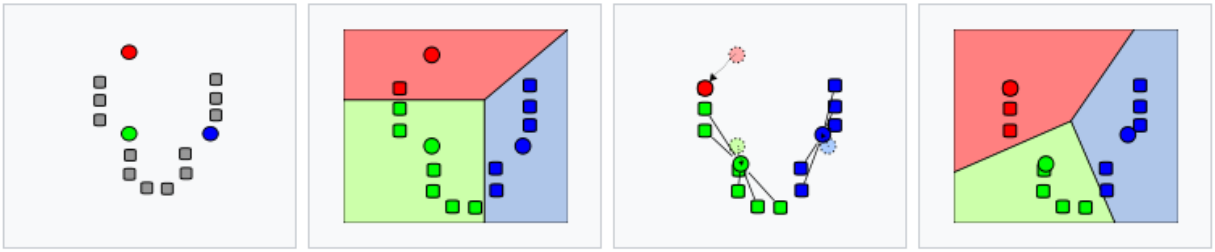
$$\sum_{i=1}^{N} d(x_i, m_j(x_i))^2 ,$$

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.

Опис алгоритму

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

1. Дослідник визначає кількість кластерів, що необхідно створити
2. Випадковим чином обирається k спостережень, які на цьому кроці вважаються центрами кластерів
3. Кожне спостереження «приписується» до одного з n 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 кожному кластері опинятимуться одні й ті самі об'єкти), дисперсія всередині кластера буде мінімізована, а між кластерами — максимізована



1. *k* початкових «середніх» (тут *k*=3) випадково згенеровані у межах домени даних (кольорові).

2. створено *k* кластерів, асоціюючи кожне спостереження з найближчим середнім. Розбиття відбувається згідно з діаграмою Вороного утвореною середніми.

3. Центроїд кожного з *k* кластерів стає новим середнім.

4. Кроки 2 і 3 повторюються до досягнення збіжності.

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

Принцип дії

Принцип алгоритму полягає в пошуку таких центрів кластерів та наборів елементів кожного кластера при наявності деякої функції Ф(°), що виражає якість поточного розбиття множини на k кластерів, коли сумарне квадратичне відхилення елементів кластерів від центрів цих кластерів буде найменшим:

$$V = \sum_{i-1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2$$

де k — число кластерів, S_i — отримані кластери, $i = 1, 2, \ldots, k$, μ_i — центри мас векторів $x_j \in S_i$.

В початковий момент роботи алгоритму довільним чином обираються центри кластерів, далі для кожного елемента множини ітеративно обраховується відстань від центрів з приєднанням кожного елемента до

кластера з найближчим центром. Для кожного з отриманих кластерів обчислюються нові значення центрів, намагаючись при цьому мінімізувати функцію Ф(°), після чого повторюється процедура перерозподілу елементів між кластерами.

Алгоритм методу «Кластеризація за схемою к-середніх»:

1. вибрати k інформаційних точок як центри кластерів поки не завершиться процес зміни центрів кластерів;
2. зіставити кожну інформаційну точку з кластером, відстань до центра якого мінімальна;
3. переконатися, що в кожному кластері міститься хоча б одна точка. Для цього кожний порожній кластер потрібно доповнити довільною точкою, що розташована «далеко» від центра кластера;
4. центр кожного кластера замінити середнім від елементів кластера;
5. Кінець.

Переваги і недоліки

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 (англ. fuzzy k-means methods), у яких на першій стадії алгоритму допускається приналежність одного елемента множини до декількох кластерів (із різним ступенем приналежності).

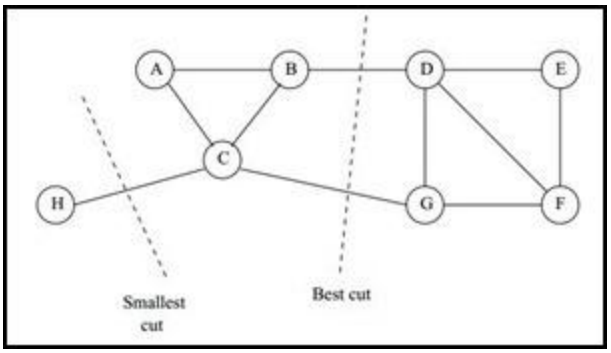
Незважаючи на очевидні переваги методу, він має суттєві недоліки:

1. Результат класифікації сильно залежить від випадкових початкових позицій кластерних центрів
2. Алгоритм чутливий до викидів, які можуть викривлювати середнє
3. Кількість кластерів повинна бути заздалегідь визначена дослідником

Метод k-середніх є доволі простим і прозорим, тому успішно використовується у різноманітних сферах — маркетингових сегментаціях, геостатистиці, астрономії, сільському господарстві тощо.

Спектральна кластеризація(Spectral clustering)

Спектральна кластеризація (SC) - це метод кластеризації, заснований на теорії графів - зважений неорієнтований граф ділиться на два або більш оптимальні підграфи, так що внутрішня частина підграфа є максимально схожою. Відстань між картами має бути максимально можливою для досягнення загальної мети кластеризації. Оптимальний серед них відноситься до іншої оптимальної цільової функції, яка може бути мінімальним розрізом ріжучої кромки, як показано на Найменшому зрізі на малюнку 1 (наприклад, мінімальний зріз нижче), або зрізом того ж розміру та мінімального зрізу, наприклад, Кращий зріз малюнка 1 (наприклад , Нормалізований зріз пізніше).



Спектральна кластеризація неорієнтованого графа поділу – Найменший зріз та Найкращий зріз

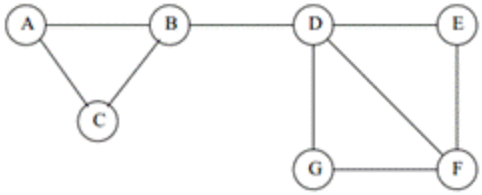
Таким чином, спектральна кластеризація може ідентифікувати вибіркові простори довільної форми і сходитися до оптимального оптимального рішення. Основна ідея полягає у використанні вибіркових даних . Матриця подоби (матриця Лапласа) . Вектори ознак, отримані після розкладання ознак, групуються.

Теоретична основа

Якщо ми обчислимо схожість між предметом і предметом, ми можемо отримати матрицю подібності тільки з предметом. Далі, розглянемо предмет як Вершину (V) у Графі (G), а подібність між піснями як G Край (E), тому ми отримуємо нашу загальну концепцію графа.

Для подання графіка (малюнок 2) зазвичай використовуються:

2-1 图



2-2 邻接矩阵

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

2-3 L矩阵

$$\begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 4 & -1 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 2 \end{bmatrix}$$

Матриця суміжності: E, eijЗначить viViВаги ребер, E - симетрична матриця, а діагональні елементи - 0, як показано на малюнку 2-2.

Лапласова матриця: L = D - E, де di (Сума елементів рядка або стовпця), як показано на малюнку 2-3.

Власні значення і L матриця.

Спочатку розглянемо оптимізований метод сегментації зображення, взявши дихотомію як приклад, розрізаючи графік на дві частини, S і T, що еквівалентно наступному розрізання функції втрат (S, T), як показано в рівнянні 1, яке є найменшим (який знаходився) Зважена сума ребер).

$$cut(S,T)=\sum_{i\in S,j\in T} e_{ij} \quad (1)$$

$$q=[q_1,q_2,\ldots,q_n]^T \quad (2)$$

$$q=\begin{cases} c_1 & i\in S \\ c_2 & i\in T \end{cases} \quad (3)$$

Припустимо, що вони розділені на дві категорії, S і T і q (як показано в рівнянні 2) представляє класифікацію, а q задовольняє співвідношенню рівняння 3, яке використовується для ідентифікації класу.

Потім:

$$cut(S,T)=\sum_{i\in S,j\in T} e_{ij}=\frac{\sum_{i=1}^{n}\sum_{j=1}^{n} e_{ij}(q_i-q_j)^2}{2(c_1-c_2)^2}$$

又:

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{n} e_{ij}(q_i-q_j)^2 &= \sum_{i=1}^{n}\sum_{j=1}^{n} e_{ij}(q_i^2-2q_iq_j+q_j^2 \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} -2e_{ij}q_iq_j+\sum_{i=1}^{n} q_i^2(\sum_{j=1}^{n} e_{ij}) \\ &= 2q^T(D-E)q \\ &= 2q^TLq\end{aligned}$$

Де D - діагональна матриця, сума елементів рядка або стовпця, а L - матриця Лапласа.

За:

$$q^TLq=\frac{1}{2}(\sum_{i=1}^{n}\sum_{j=1}^{n} e_{ij}(q_i-q_j)^2)$$
$$q^Tq=n$$

Є:

1. L - симметрична позитивна полуопределенная матриця, яка гарантує, що всі власні значення більше або дорівнює 0;

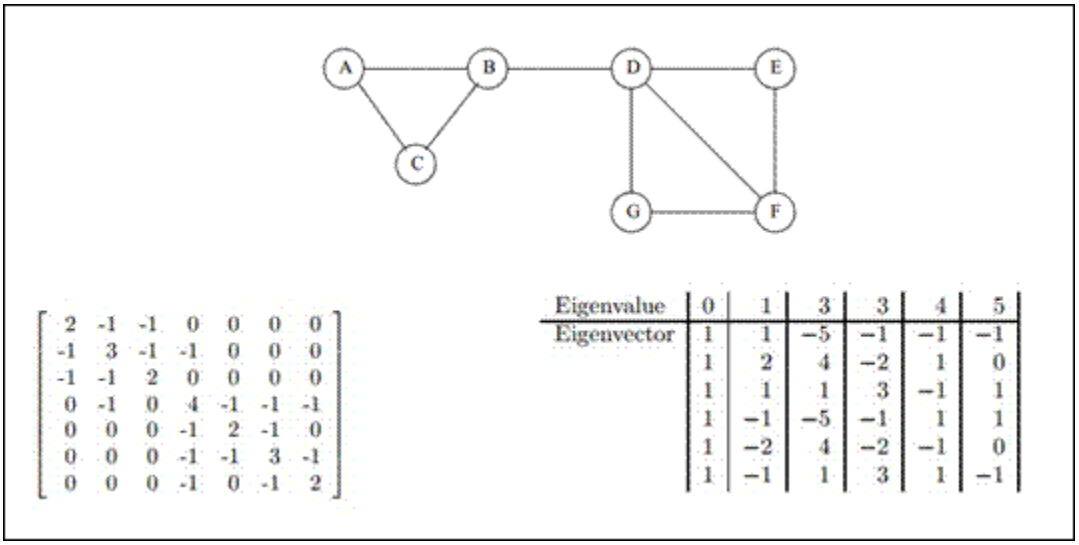
2. Матриця L має унікальне власне значення 0, і відповідний їй власний вектор дорівнює 1.

Важко вирішити q дискретно. Якщо завдання пом'якшена у безперервні реальні значення, природа ентропії Релея знає, що мінімальне значення вашого типу - це власні значення L (мінімальне значення, друге найменше значення, ..., максимальне значення відповідає мінімальному власним значенням матриці L, другого найменшим власним значенням, ..., максимальному власному значенню і власного вектора, що відповідає екстремальному значенню q, див. ентропію Релея (Релея quotient))。

На цьому етапі треба перетворити розріз (S, T) у задачу про власні значеннях (векторах) матриці Лапласа і послабити дискретну проблему кластеризації в безперервні власні вектори. Найменший вектор ознак серії відповідає оптимальному методом ділення на ряди графа. Решта просто для дискретизації проблеми релаксації, тобто для повторного розподілу векторів ознак ви можете отримати відповідні категорії. Наприклад, найменший вектор ознак на малюнку 3 ділиться на позитивну і негативну, і ви отримуєте клас {A, B , C} і класи {D, E, F, G}. В K класифікації перші K векторів ознак часто класифікуються за kmeans.

PS:

1. Хоча k means згадується тут знову, значення далеко від kmeans, що обговорювався, коли була представлена концепція.Kmeans тут більше пов'язані з вивченням ансамблів і тут не описуються;
2. Число k і кількість кластерів не збігаються, і це можна зрозуміти з відповідного фізичного значення.
3. Серед перших k власних векторів значення в першому стовпці в точності збігаються (це значення дуже схоже, коли ітераційний алгоритм обчислює власні вектори). Його можна видалити, коли k means, і його також можна використовувати для простого визначення методу власних значень (вектора) стовпця. Правильно це чи ні, часто з-за асиметричної матриці суміжності.



$$\begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 4 & -1 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 2 \end{bmatrix}$$

Eigenvalue	0	1	3	3	4	5
Eigenvector	1	1	−5	−1	−1	−1
	1	2	4	−2	1	0
	1	1	1	3	−1	1
	1	−1	−5	−1	1	1
	1	−2	4	−2	−1	0
	1	−1	1	3	1	−1

Метод оптимізації

В інших методах кластеризації, таких як k means, складно охарактеризувати співвідношення розмірів класів, і локальне оптимальне

рішення також є неминучою витоком. Звичайно, це виключно для широкого використання k means - принцип простий.

1. Метод мінімальної різання

Як і в методі розрахунку в розділі 2.2, оптимальна цільова функція виглядає наступним чином:

$$
\begin{aligned}
&cut(S,T) = \sum_{i\in S, j\in T} e_{ij} \\
&\qquad = q^T L q \\
&s.t.: q^T q = n; \quad 源于 q_i = 1(-1); \\
&\qquad [1,1,...,1]^T q = 0; 源于特征值0对应的特征向量为1.
\end{aligned}
$$

Метод розрахунку може бути безпосередньо вирішене шляхом обчислення мінімального власного значення (власного вектора) L.

2. Номеризованный метод різання

Нормалізований розріз, мета полягає в тому, щоб одночасно розглянути мінімізований розріз, розділити ваги, щоб не вирізати жодної букви H, як показано на малюнку 1. Критерієм для вимірювання розміру під-зображення є: сума ступенів кінцевих точок під-зображення.

$obj = cut(S,T) * (\frac{1}{d_1} + \frac{1}{d_2}) = (\frac{1}{d_1} + \frac{1}{d_2}) \sum_{i \in S, j \in T} e_{ij}$; d_1为图1的权值和，n_2为图2的权值和

$$= \sum_{i \in S, j \in T} e_{ij}(q_i - q_j)^2; \quad q_i = \begin{cases} \sqrt{\frac{d_1}{d_2 d}} \\ -\sqrt{\frac{d_2}{d_1 d}} \end{cases}$$

$$= q^T L q$$

$$s.t.: q^T D q = 1;$$

$$q^T D q = 0;$$

泛化瑞利熵为：$\frac{q^T L q}{q^T \mathrm{D} q}$。

而：

$$\mathrm{L} q = \lambda D q$$

$$\Leftrightarrow \mathrm{L} q = \lambda D^{\frac{1}{2}} D^{\frac{1}{2}} q$$

$$\Leftrightarrow D^{-\frac{1}{2}} L D^{-\frac{1}{2}} D^{-\frac{1}{2}} q = \lambda D^{-\frac{1}{2}} q$$

$$\Leftrightarrow L' q' = \lambda q'$$

其中$\mathrm{L} = D^{-\frac{1}{2}} \mathrm{L} D^{-\frac{1}{2}}, q = D^{-\frac{1}{2}} q.$

因而只需要将原L矩阵，归一化即可，$\mathrm{L} = D^{-\frac{1}{2}} \mathrm{L} D^{-\frac{1}{2}}$。

3. Метод перерізу відносини

Співвідношення вирізу. Мета полягає в тому, щоб одночасно розглянути можливість мінімізації країв реза і ділення балансу, щоб не вирізати жодного H, подібного показаному на рисунку.

Оптимальна цільова функція:

$$obj = cut(S,T) * (\frac{1}{n_1} + \frac{1}{n_2}) = (\frac{1}{n_1} + \frac{1}{n_2}) \sum_{i \in S, j \in T} e_{ij}$$；n_1为图1的顶点和，n_2为图2的顶点和

$$= \sum_{i \in S, j \in T} e_{ij}(q_i - q_j)^2$$

$$= q^T L q$$

$$s.t.: q^T q = 1;$$

$$[1,1,...,1]^T q = 0;$$

计算方式和$\min cut$类似，由瑞利熵求取最小的特征值

4. Нормалізоване перетворення подібності

Нормалізована матриця L:

$$L' = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} E D^{-\frac{1}{2}}$$

又：

$$(I - D^{-\frac{1}{2}} E D^{-\frac{1}{2}})x = \lambda x \Leftrightarrow (D^{-\frac{1}{2}} E D^{-\frac{1}{2}})x = (1-\lambda)x$$

Таким чином, L' – найменше власне значення і D-(1/2)E D-(1/2) відповідає найбільшому власному значенню.

І розрахована л порівняно з обчисленням L він має невелику перевагу. У практичних додатках L часто використовується. Замініть L, але мінімальне скорочення і скорочення відносини не допускаються.

PS: Це також часто зустрічається в блогах людей: А каже, що спектральна кластеризація призначена для знаходження найбільшого K власного значення (вектора), а Б говорить, що спектральна кластеризація призначена для знаходження найменшого K власного значення (причини вектора).

3 спектральних етапи кластеризації

Перший крок: підготовка даних, генерація матриці суміжності графа;

Другий крок: нормалізована плазмова матриця;

Крок 3: Генерація найменших k значень ознак і відповідних векторів ознак;

Четвертий крок: кластеризація векторів ознак kmeans (невелика кількість векторів ознак);

Фізична значимість спектральної кластеризації

Матриця в спектральній кластеризації:

邻接矩阵：

$$E=\begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ e_{n1} & e_{n1} & e_{n1} & e_{n1} \end{bmatrix}$$

min *cut*和*ratiocut*中的Laplacian矩阵：

$$L=D-E \Leftrightarrow -\mathrm{L}=E-D$$

*Normalizedcut*中的L′：

$$\mathrm{L}'=D^{-\frac{1}{2}}\mathrm{L}D^{-\frac{1}{2}}=I-D^{-\frac{1}{2}}ED^{-\frac{1}{2}}$$

Мабуть це L, L'Обидва особливо пов'язані з E. Якщо E розглядається як багатовимірне векторний простір, воно також може певною мірою відображати відносини між елементами. Кластеризація E безпосередньо з допомогою kmeans, отримані результати можуть також відображати характеристики кластеризації V і введення спектральної кластеризації L і L'Це робить поділ G фізичним значенням. Більш того, якщо елемент E (Тобто

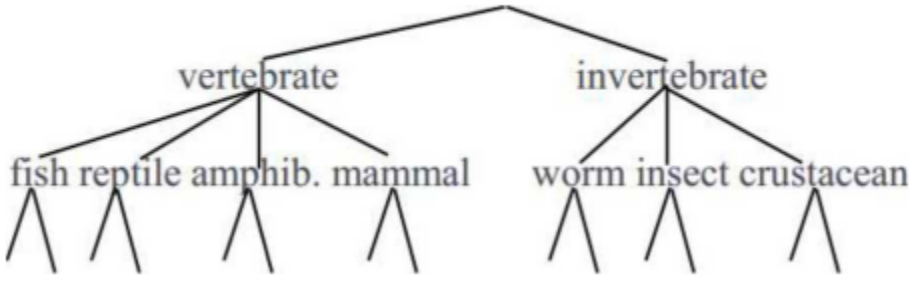
N) досить велика, буде складно розрахувати його kmeans, ми можемо використовувати PCA, щоб зменшити розмірність (все ще верхні власні значення і вектори).

Вищезгадана пара розглядає E як матрицю векторного простору, яка інтуїтивно виглядає узгодженої з нашим пізнанням, але не має теоретичної основи, в той час як L (L'Введення і т. Д., Як описано в розділі 2, робить розрахунок теоретичним підґрунтям, і перші k власних векторів також еквівалентні L (L'I т. д).

Отже, кластеризація полягає в тому, щоб знайти теоретичну основу для поділу графів, яка може досягти мети зменшення розмірності.

Ієрархічна (Агломеративна) кластеризація

Завдання ієрархічної кластеризації — побудувати ієрархію кластерів



Ієрархія будується автоматично:

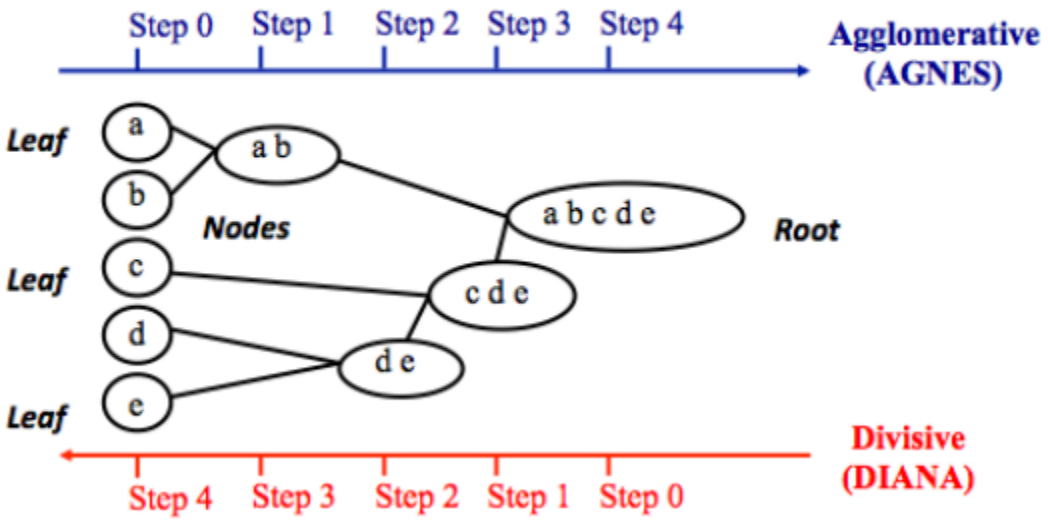
1. або згори-вниз (агломеративні алгоритми) - AGNES (AGglomerative NESting): ROCK, CURE,CHAMELEON

2. або знизу-вгору (алгоритми розділення) - DIANA(DIvisive ANAlysis): BIRCH, MST

Як в агломеративній, так і в роздільній ієрархічній кластеризації, користувачам потрібно вказати бажану кількість кластерів та умови завершення.

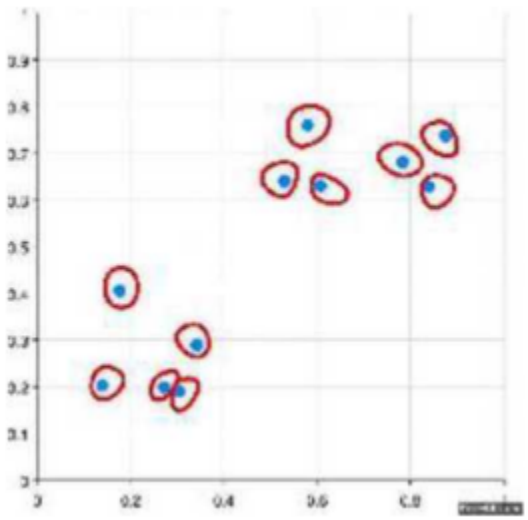
На початку роботи агломеративного алгоритму кожна точка розглядається як кластер, потім алгоритм намагається об'єднати найближчі сусідні точки в один більший кластер і так далі, щоб зрештою об'єднати всі кластери в один великий кластер

Алгоритм розділення спочатку розглядає всі точки множини як один кластер; на подальших кроках деякі кластери вищого рівня рекурсивно розщеплюються для побудови діаграми. Ці підходи протилежні один одному.



Найвідоміший метод побудови знизу-вгору: ієрархічна агломеративна кластеризація.

- Будує ієрархію у вигляді двійкового дерева
- Використовує міру близькості для визначення подібності двох кластерів



Алгоритм:

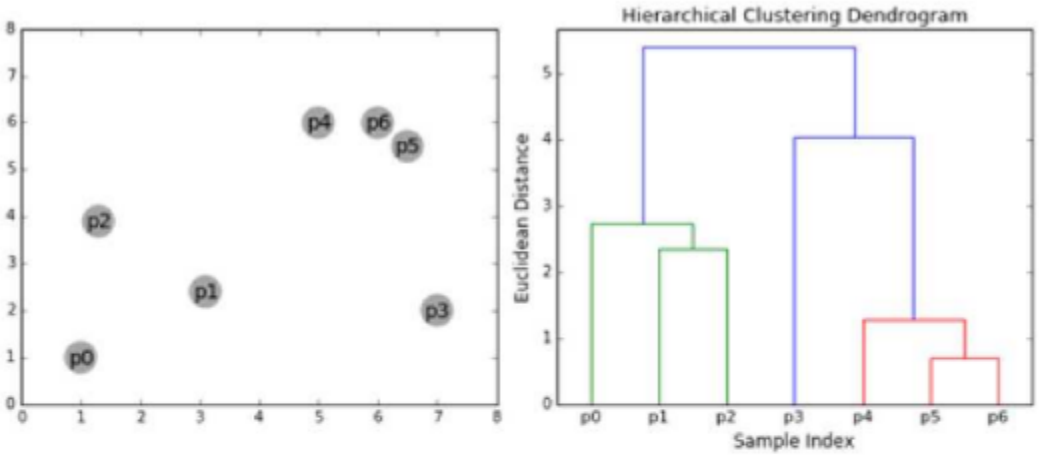
1. Спочатку кожен об’єкт розглядається як окремий кластер
2. По черзі об'єднуємо два найбільш схожих кластера До тих пір поки не залишиться один кластер
3. Історія об'єднань формує дерево ієрархії
4. Така історія зображується дендограмою

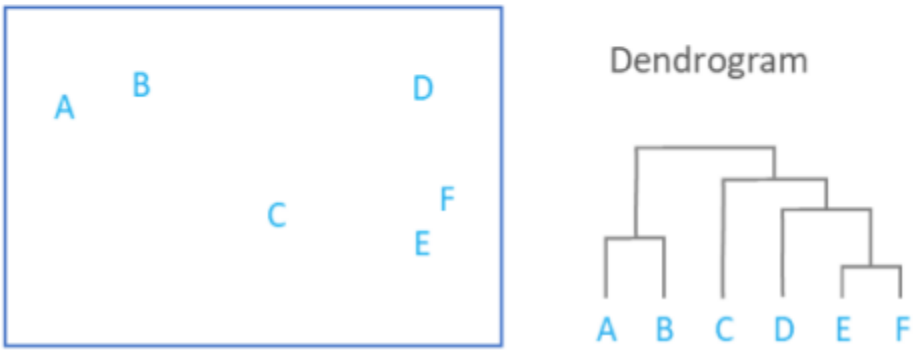
Дендограма

Дендрограма - це тип деревної діаграми, що показує ієрархічні взаємозв'язки між різними наборами даних.

Дендрограма містить пам'ять ієрархічного алгоритму кластеризації, тому, просто переглянувши дендрограмму, ви можете сказати, як формується

кластер. Відстань між точками даних означає несхожість. Висота блоків представляє відстань між кластерами





Обчислювальна складність ієрархічної кластеризації

1. Обчислюємо близькість всіх NxN пар об’єктів
2. Потім, на кожній ітерації:
    a. Скануємо O(NxN) близькостей для знаходження максимальної
    b. Об'єднуємо два кластери
    c. Обчислюємо близькість між створеним кластером і всіма рештою

Всього O(N) ітерацій, кожна вимагає O(NxN) сканувань. Загальна складність: O(N3). Існує більш раціональна модифікація алгоритму зі складністю O(N2).

Таблиці і діаграми, в яких зібрано результати дослідження. Алгоритм роботи

Процесинг даних

Проблема, описана в цьому наборі даних, вимагає від нас виділення сегментів клієнтів в залежності від моделей їх поведінки, представлених в наборі даних, щоб сфокусувати маркетингову стратегію компанії на конкретному сегменті.

Давайте спочатку завантажимо набір даних і швидко поглянемо на нього, щоб визначити наш підхід до вирішення цієї проблеми.

CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOFF_PURCHASES_FREQUENCY	PURCHASES_INSTALLMENTS_FREQUENCY	CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_TRX	PURCHASES_TRX	CREDIT_LIMIT	PAYMENTS	MINIMUM_PAYMENTS	PRC_FULL_PAYMENT	TENURE	
0	C1000	40.900749	0.818181	95.40	0.00	95.4	0.000000	0.166667	0.000000	0.083333	0.000000	0	2	1000	201.802084	139.509787	0.000000	12

	01		82											.0			00	
1	C100002	3202.467416	0.909091	0.00	0.00	0.0	6442.945483	0.000000	0.000000	0.000000	0.250000	4	0	700000.0	4103.032597	1072.3402217	0.22222222	12
2	C100003	2495.148862	1.000000	773.17	773.17	0.0	0.000000	1.000000	1.000000	0.000000	0.000000	0	12	750000.0	622.066742	627.2847877	0.0000000	12
3	C100004	1666.670542	0.636364	1499.00	1499.00	0.0	205.788017	0.083333	0.083333	0.000000	0.083333	1	1	750000.0	0.000000	NaN	0.0000000	12
4	C100005	817.714335	1.000000	16.00	16.00	0.0	0.000000	0.083333	0.083333	0.000000	0.000000	0	1	120000.0	678.334763	244.791237	0.0000000	12

	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	ONEOFF_PURCHASES_FREQUENCY	PURCHASES_INSTALLMENTS_FREQUENCY	CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_TRX	PURCHASES_TRX	CREDIT_LIMIT	PAYMENTS	MINIMUM_PAYMENTS	PRC_FULL_PAYMENT	TENURE	
count	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8950.000000	8949.000000	8950.000000	8637.000000	8950.000000	8950.000000	8950.000000
mean	1564.474828	0.877271	1003.204834	592.437371	411.067645	978.871112	0.490351	0.202458	0.364437	0.135144	3.248827	14.709832	4494.449450	1733.143852	864.206542	0.153715	11.517318	
std	2081.5	0.236	2136.63	1659.887917	904.33	2097.16	0.401371	0.298336	0.397448	0.200	6.82	24.85	3638.	2895.06	2372.44	0.292	1.1338	

	31879	904	4782		8115	3877				121	4647	7649	8157225	3757	6607	499	331
min	0.000000	0.0000000	0.0000000	0.000000	0.0000000	0.000000	0.000000	0.000000	0.000000	0.0000000	0.0000000	0.0000000	50.0000000	0.0000000	0.019163	0.0000000	6.0000000
25%	128.281915	0.8888889	39.6350000	0.000000	0.0000000	0.000000	0.083333	0.000000	0.000000	0.0000000	0.0000000	1.0000000	1600.0000000	383.276166	169.123707	0.0000000	12.000000
50%	873.385231	1.0000000	361.2800000	38.000000	89.000000	0.000000	0.500000	0.083333	0.166667	0.0000000	0.0000000	7.0000000	3000.0000000	856.901546	312.343947	0.0000000	12.000000

75%	2054.140036	1.000000	1110.130000	577.405000	468.637500	1113.821139	0.916667	0.300000	0.750000	0.222222	4.000000	17.000000	6500.000000	1901.134317	825.485459	0.142857	12.000000
max	19043.138560	1.000000	49039.570000	40761.250000	22500.000000	47137.211760	1.000000	1.000000	1.000000	1.500000	123.000000	358.000000	30000.000000	50721.483360	7640.620752	1.000000	12.000000

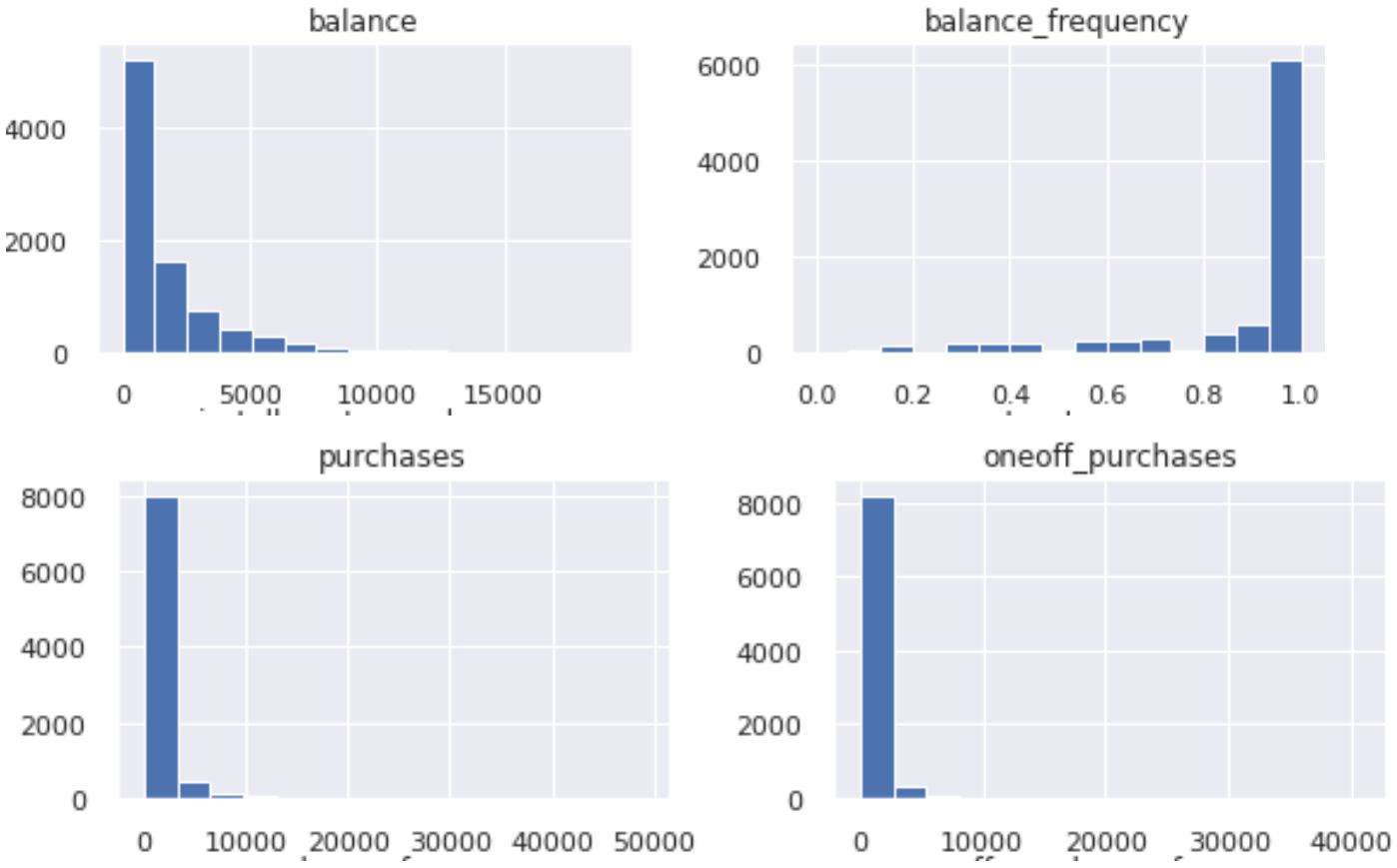
Переглядаючи звіти, ми бачимо, що середнє значення більшості функцій набагато перевищує їх медіану. Це ознака деякої асиметрії в наборі даних, і ми повинні подивитися, чи можемо ми щось з цим зробити. У нас також є деякі значення Nan для розміщення там.

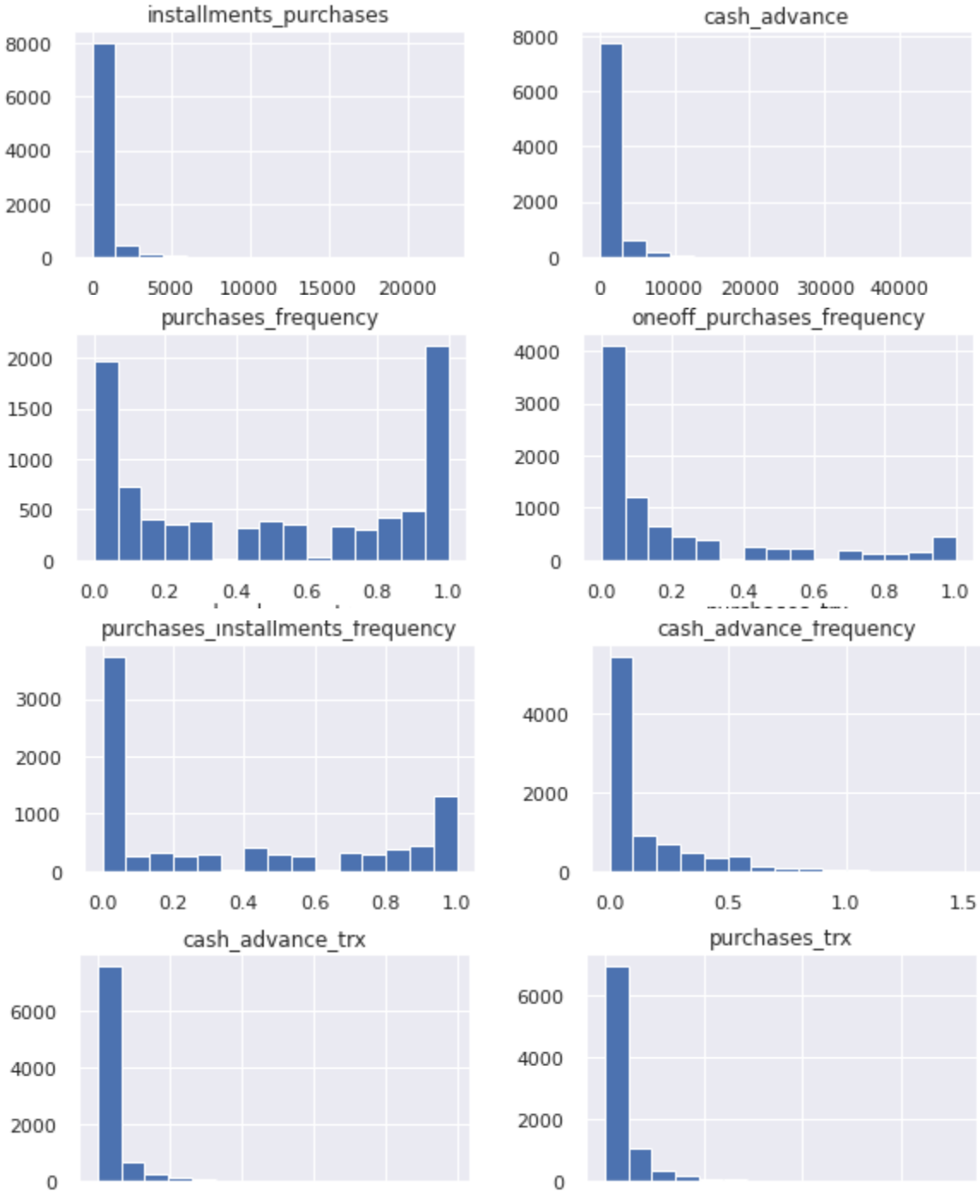
Ідентифікатор клієнта, мабуть, є унікальним ідентифікатором для кожного клієнта і, отже, не буде грати ніякої ролі у визначенні кластера.

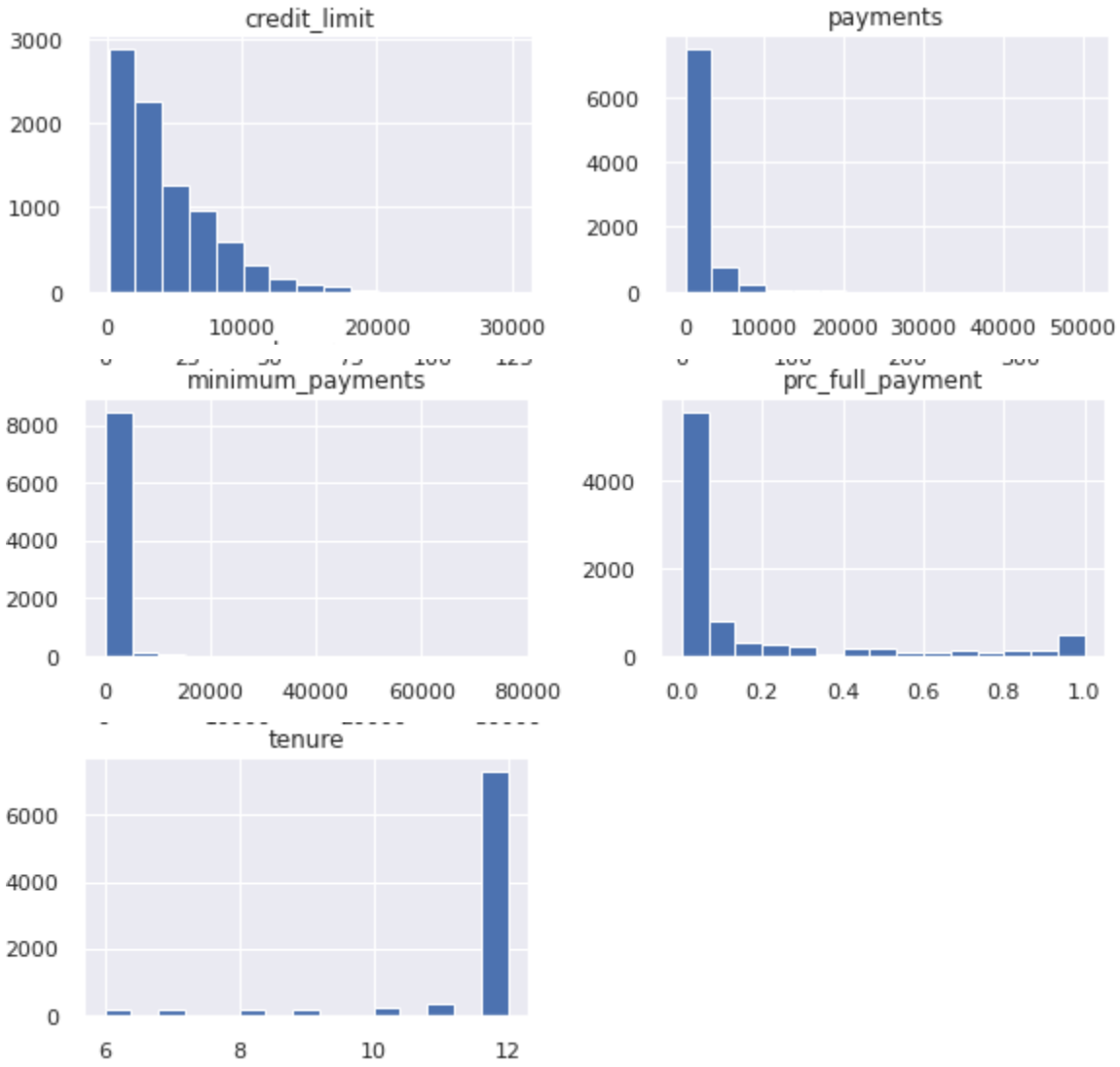
Ми бачили, що функція кредитного ліміту містить всього 0,01% записів, що мають значення Nan, тобто тільки 1 запис тут має відсутнє значення. Так що нам не потрібно обтяжувати себе тим, щоб ставити це в провину. Ми можемо просто відкинути це і ніколи більше не думати про це.

Що стосується функції подання мінімальних платежів, я не бачу жодного стовпця, який мав би відношення до цієї функції і допомагав би нам оцінити значення для відсутніх записів. Здається, що значення відсутні випадковим чином, і ми можемо просто використовувати медіану для заміни значень Nan, оскільки розподіл для мінімальних платежів спотворено, і, отже, медіана дає кращу оцінку центральної тенденції цієї функції.

Візуалізація даних



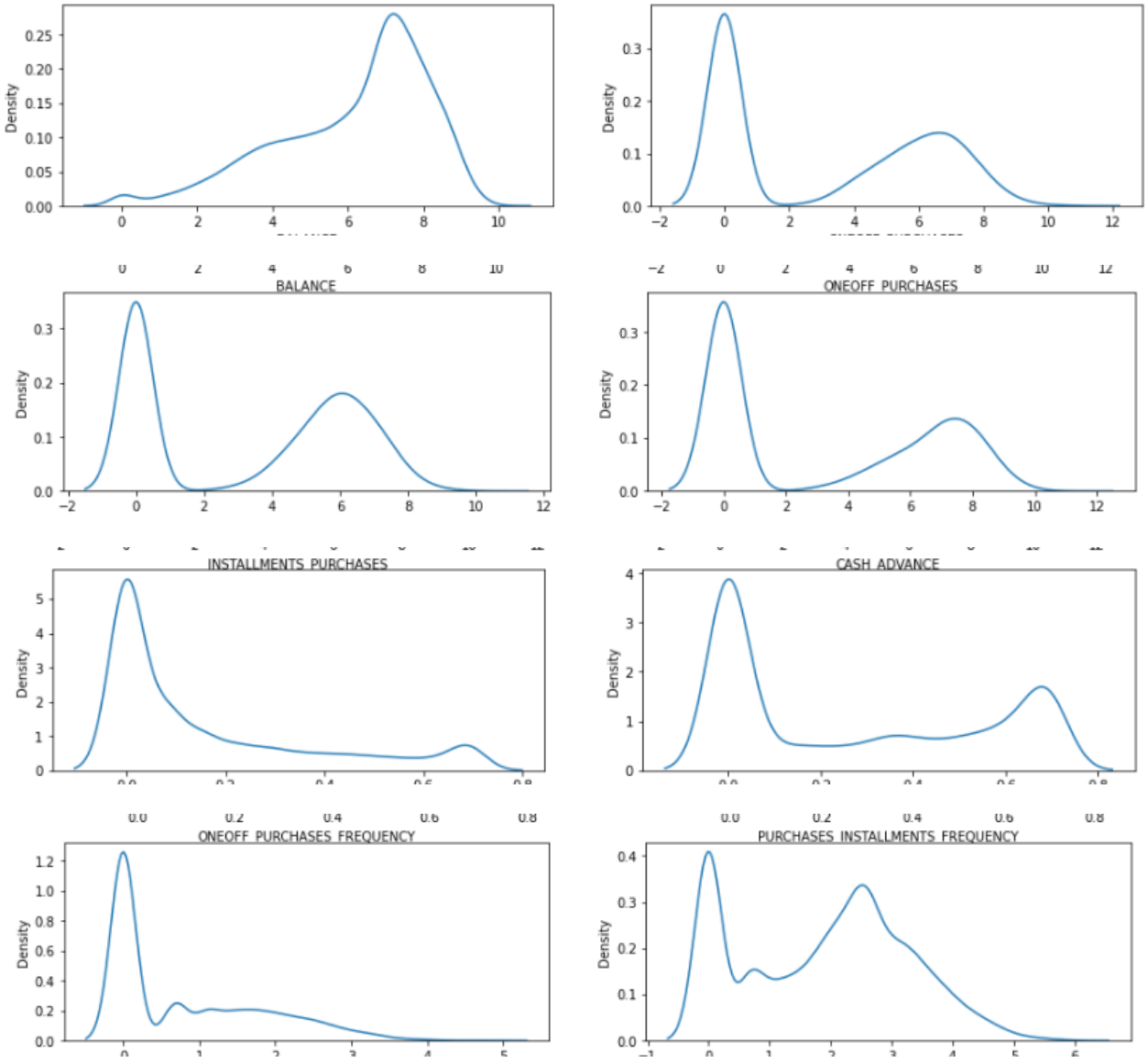

installments_purchases
cash_advance
purchases_frequency
oneoff_purchases_frequency
purchases_installments_frequency
cash_advance_frequency
cash_advance_trx
purchases_trx

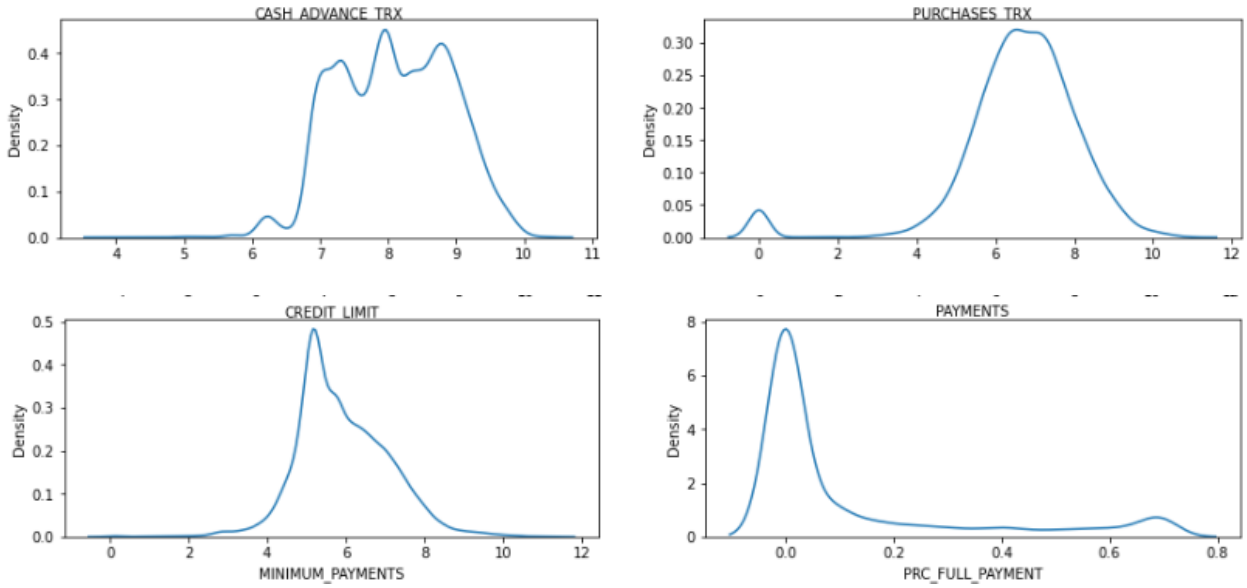


Тут багато перекосів, і вони різноманітні. Це частково очікувано від подібних наборів даних, оскільки завжди знайдеться кілька клієнтів, які здійснюють дуже велику кількість транзакцій.

Тепер від програми залежить, чи хочемо ми обробляти асиметрію в нашому наборі даних чи ні для проблеми кластеризації. Наприклад, якщо ми хочемо виконати кластеризацію для виявлення аномалій, в цьому випадку ми не хотіли б обробляти викиди, оскільки ми хотіли б, щоб наша модель виявляла їх і групувала в кластер. Для нашого застосування я шукаю хорошу

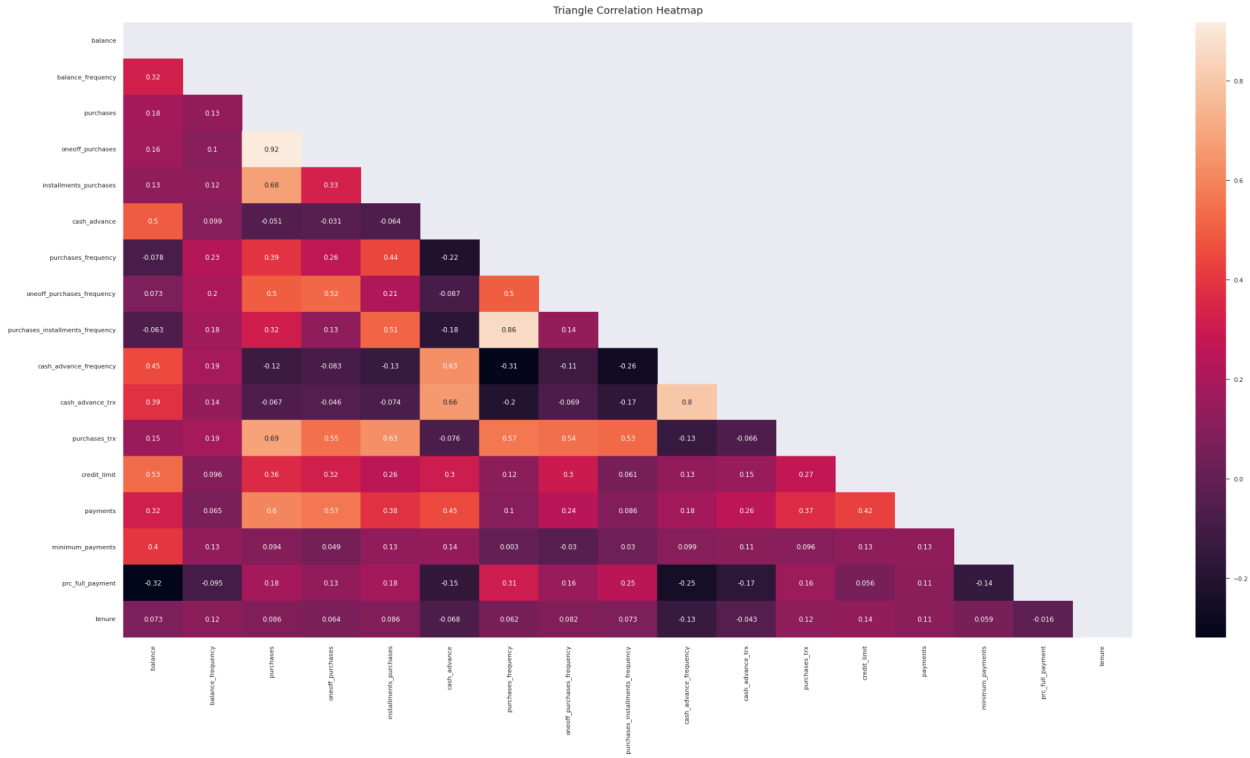
візуалізацію, тому я хотів би максимально усунути асиметрію, оскільки це допоможе моделі формувати кращі кластери. Давайте подивимося, чи зможемо ми щось з цим зробити.

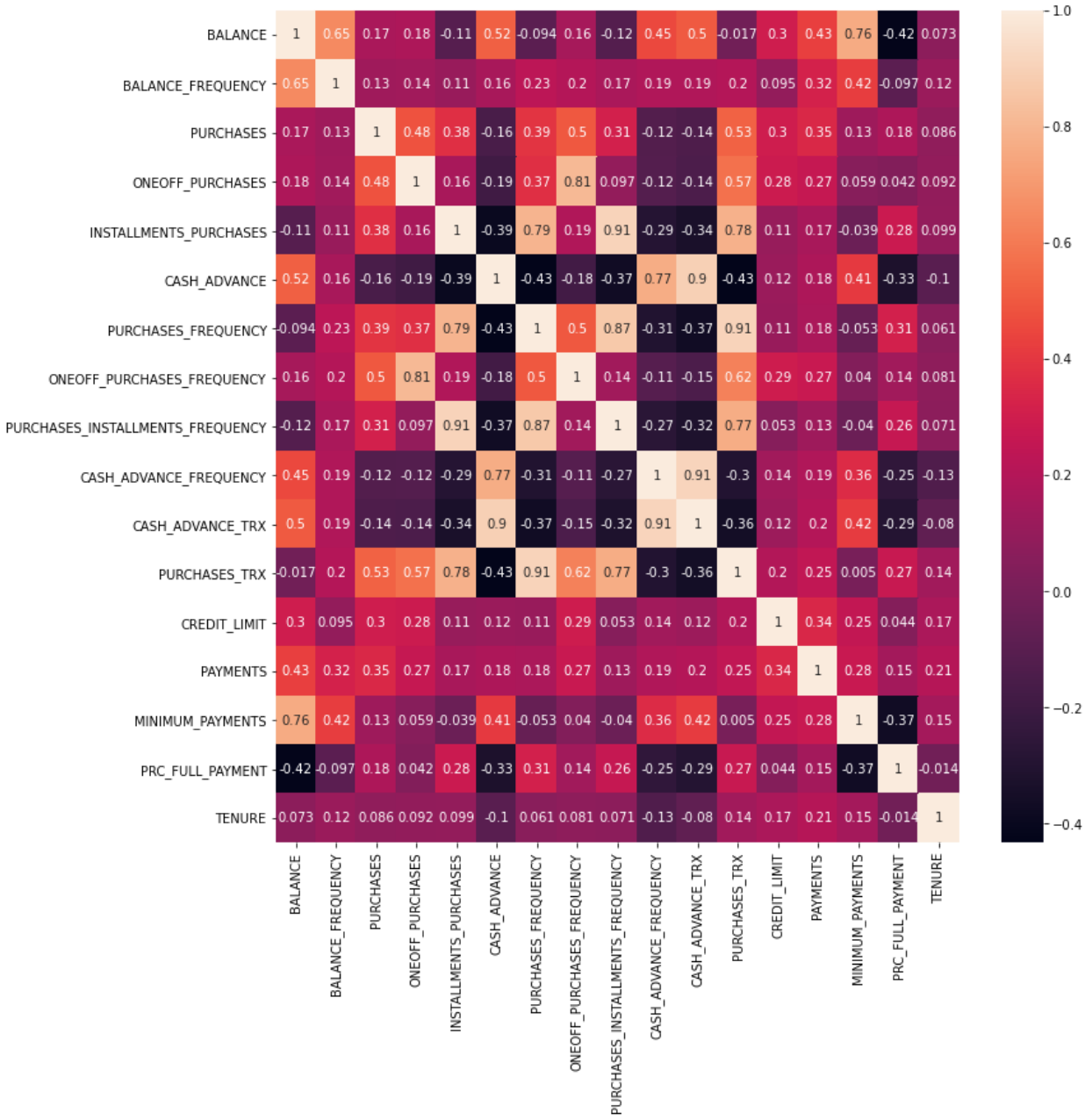




Я знаю, що це може здатися не ідеальним розподілом, але це краще, ніж те, що ми мали, і наше завдання як фахівця з обробки даних - максимально допомогти нашій моделі.

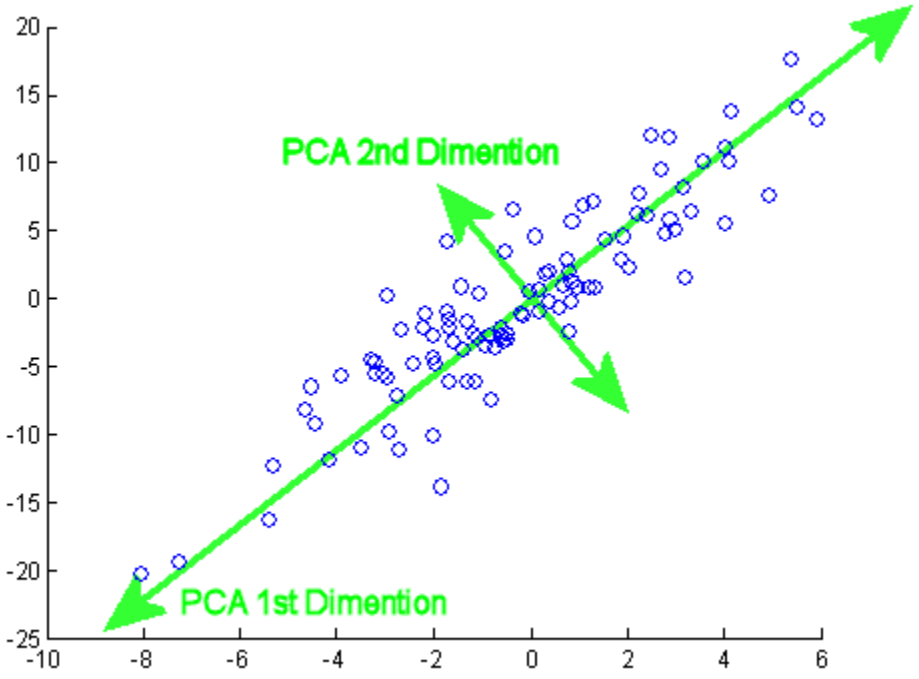
Зараз подивимося на матрицю кореляцій до і після:





Тут у нас є кілька взаємопов'язаних функцій. Є багато способів впоратися з цим. Ми продовжимо роботу по зменшенню розмірності і приведемо ваші дані до більш низького розміру. Ми будемо використовувати PCA для нашого зменшення розмірності.

Щоб коротко пояснити, що робить PCA під капотом, він знаходить новий вимір / вісь для набору даних таким чином, щоб він пояснював максимальну дисперсію. Тоді ця вісь є першим головним компонентом. Потім він вибирає інший компонент, перпендикулярний першому головному компоненту, який пояснює максимальну дисперсію.



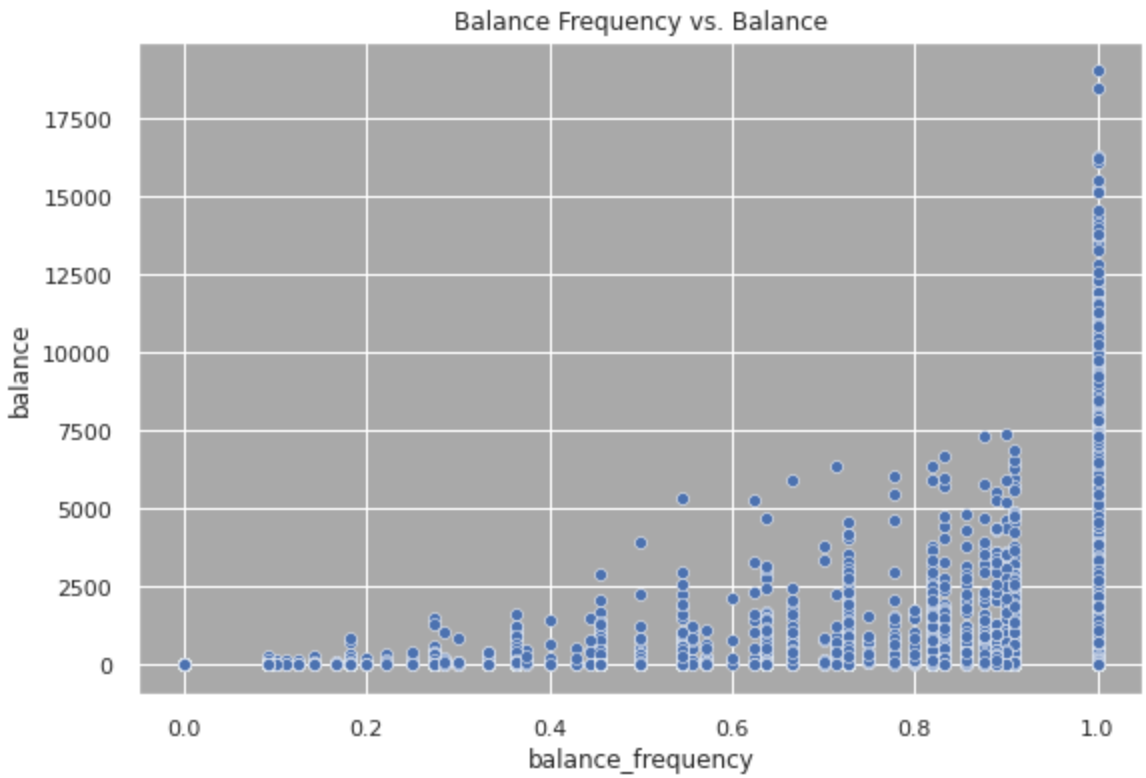
Отже, для наведеного вище зображення, якщо ми проектуємо всі точки на 1-е вимірювання PCA, то точки будуть більш розподілені, ніж якщо б ми робили це на будь-якій іншій осі. Це означає, що 1-е вимірювання PCA пояснює максимальну дисперсію і, отже, є нашим 1-м основним компонентом. Тепер ми розглядаємо компоненти, які перпендикулярні цьому компоненту, і оскільки ці дані є 2-мірними, у нас є тільки 1 компонент, перпендикулярний нашому 1-му головному компоненту, і він стає нашим 2-м головним компонентом.

Примітка: Якщо б у нас було 3-е вимірювання, що виходить з екрану, то у нас було б 2 компоненти, перпендикулярних нашим 1-м основним компонентам, і нам довелося б вибрати той, який пояснює максимальну дисперсію двох.

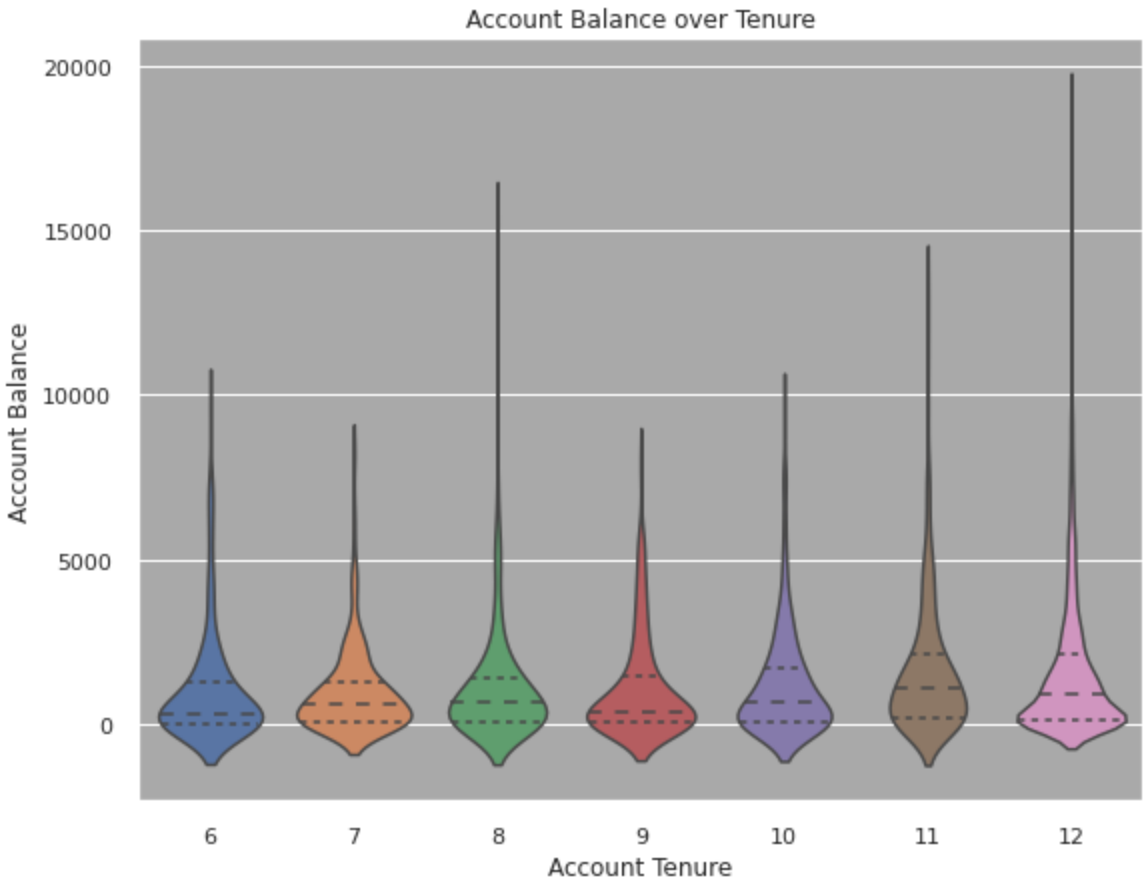
Як тільки у нас є ці основні компоненти, ми можемо вибрати кількість компонентів, які ми хочемо мати, а потім висловити наші дані в термінах цих основних компонентів і, таким чином, зменшити розміри.

Отже, давайте зробимо те ж саме в нашому випадку. Ми виберемо таку кількість компонентів, щоб наші дані в нижніх вимірах пояснювали 95% дисперсії наших вихідних даних.

З цим ми готові зробити те, що ми завжди хотіли зробити, тобто кластеризацію. Ми будемо використовувати алгоритм кластеризації K Means для вилучення кластерів інформації з нашого набору даних.


Balance Frequency vs. Balance
balance
17500
15000
12500
10000
7500
5000
2500
0
0.0
0.2
0.4
0.6
0.8
1.0
balance_frequency


Credit Limit vs. Balance
balance
17500
15000
12500
10000
7500
5000
2500
0
0
5000
10000
15000
20000
25000
30000
credit_limit


Account Balance over Tenure
20000
15000
10000
5000
0
Account Balance
6
7
8
9
10
11
12
Account Tenure

Тренування

Досить цікаво і інтуїтивно зрозуміло дізнатися, як насправді працює алгоритм кластеризації K Means.

Кластеризація K означає неконтрольований алгоритм кластеризації, який групує схожі дані в одному кластері, утворюючи k кластерів. В результаті ми отримуємо групи схожих записів, які потім можуть бути позначені і оброблені відповідним чином.

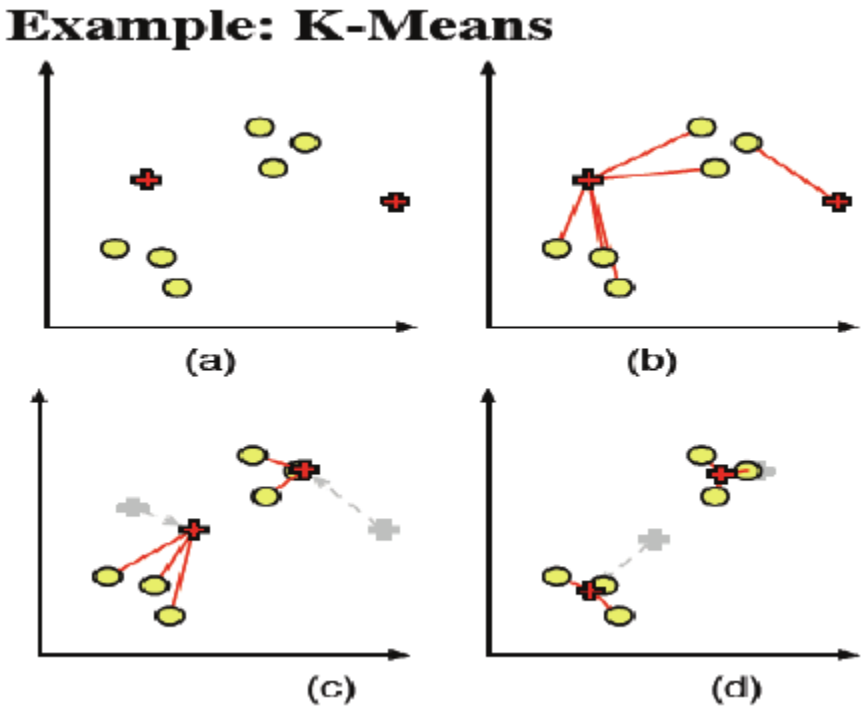
Як алгоритм знаходить кластери?

Враховуючи кількість кластерів k, він спочатку вибирає k випадкових точок (які можуть не бути точками в наборі даних) в якості k центроїдів кластера.

Потім ми присвоюємо кожній точці найближчий Центроїд і формуємо k кластерів.

Як тільки всі точки призначені кластеру, ми потім обчислюємо нові центроїди для кожного кластера. Потім ми перепризначаємо точки на найближчий Центроїд.

Якщо на кроці 4 відбувається будь-яке перепризначення, ми повторюємо кроки 3 і 4. Якщо перепризначення не відбувається, то наша модель готова, і ми витягли k кластерів з нашого набору даних. Цей процес описаний на наведеній нижче діаграмі



Однак тут є одна заковика. Оскільки сам алгоритм починається з випадкової ініціалізації k точок, багато що залежить від цієї ініціалізації. Оскільки наші дані в реальному світі розділені не так добре, як на наведеному вище малюнку, може трапитися так, що модель ініціалізує k точок таким чином, що ми можемо отримати неоптимальне рішення.

Щоб уникнути такого випадку, ми можемо запустити алгоритм кілька разів з випадковими початковими точками для кожної ітерації. Багаторазовий запуск моделі гарантує, що принаймні в одному випадку ми уникнемо невдалої ініціалізації і досягнемо оптимального рішення. Scikit learn фактично навчає модель KMeans 10 разів, і нею можна керувати за допомогою гіперпараметра if n_init.

Щоб виміряти, яка модель працює краще при n випадкової ініціалізації, ми можемо використовувати model innertia або wcss (в рамках кластерного підсумовування квадратів). Він вимірює суму відстаней кожної точки від її центру ваги. Тому нам потрібні компактні кластери з точкою якомога ближче до її центроїду.

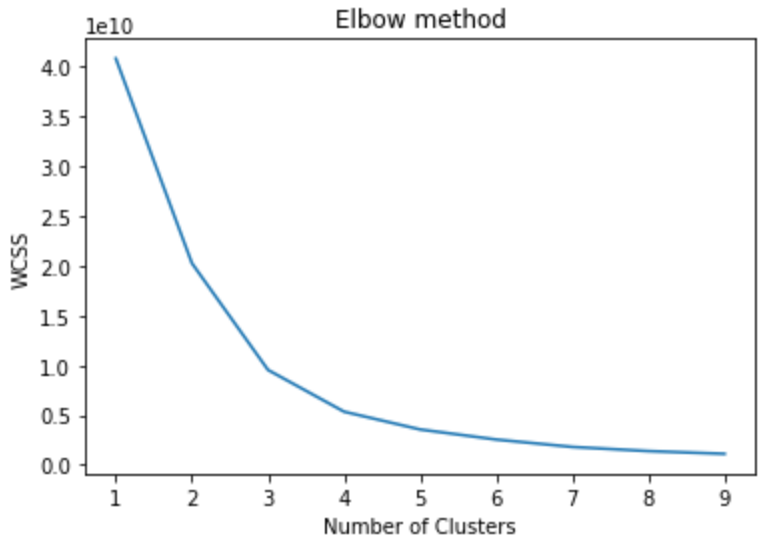
Іншим підходом, що дозволяє уникнути поганої ініціалізації, є використання алгоритму KMeans++ для ініціалізації центроїдів. Цей алгоритм ініціалізує центроїди таким чином, щоб вибрані центроїди знаходилися якнайдалі один від одного, що гарантує відсутність неоптимального рішення. Цей алгоритм поряд з попереднім підходом гарантує, що ми отримаємо найкраще можливе рішення. Scikit learn використовує алгоритм KMeans++ для ініціалізації центроїдів і задається гіперпараметром init.

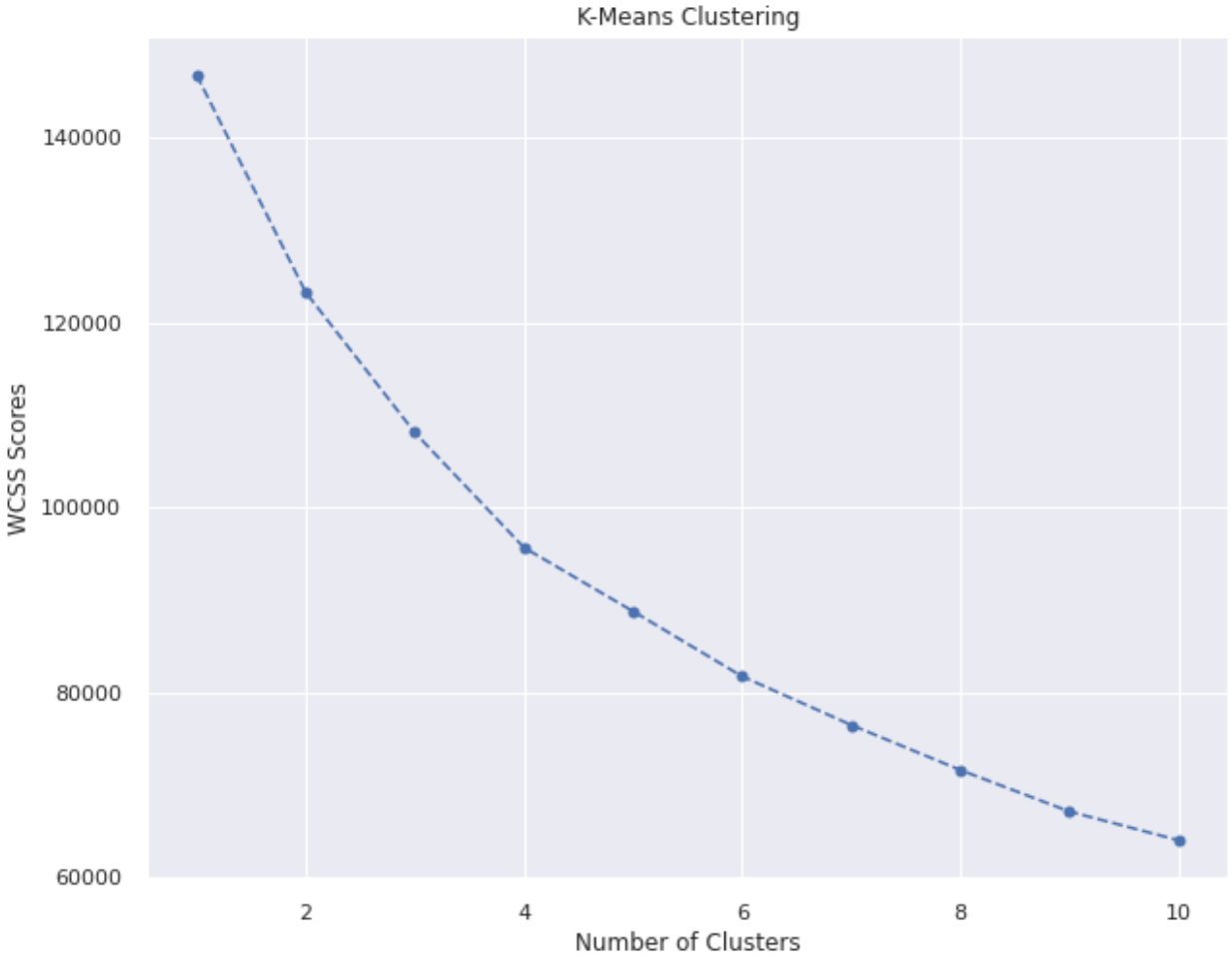
Все це обговорення до цього моменту обертається навколо припущення, що ми знаємо кількість кластерів n. Таким чином, n тут є найбільш важливим гіперпараметром, і необхідно ініціалізувати n до його відповідного значення.

Щоб отримати значення n, ми не можемо використовувати innertia як нашу метрику, так як innertia продовжує збільшуватися в міру збільшення кількості кластерів. Подумайте про це, якщо ми ініціалізуємо n, то кількість точок у наборі даних innertia буде мінімальною.

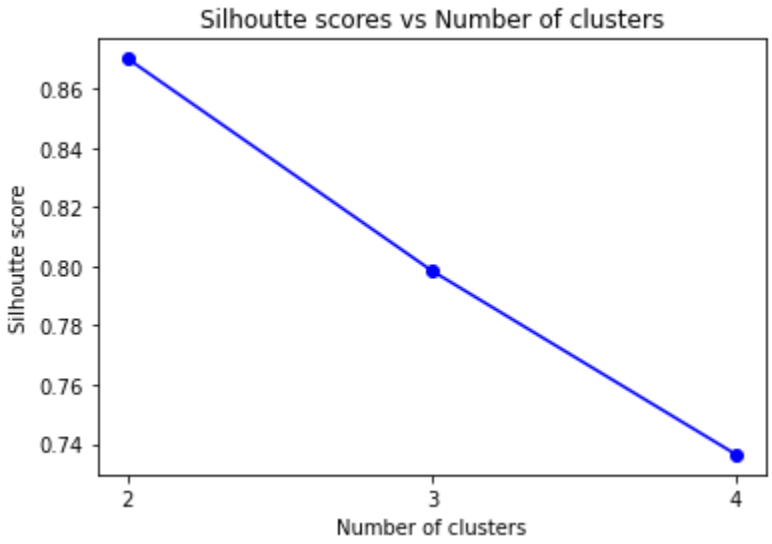
Одним із способів може бути побудова графіка залежності n від іннерції, і в міру побудови графіка ми можемо визначити лікоть, після якого іннерція зменшується з набагато меншою швидкістю. Якщо ми можемо використовувати n, що відповідає цій точці ліктя, як наше число кластерів.

Інший підхід полягає в обчисленні оцінки силуету, яка задається як (b-a) / min(a, b), де b -> середня відстань до екземплярів найближчого кластера, a -> середня відстань до інших екземплярів того ж кластера. Таким чином, оцінка силуету карає модель за те, що середня відстань до точок іншого кластера зменшується, а середня відстань до точок того ж кластера збільшується. У той час як це винагороджує модель, якщо середня відстань до точок іншого кластера збільшується, а середня відстань до точок того ж кластера зменшується. Таким чином, ми можемо вибрати модель з найвищим показником силуету.

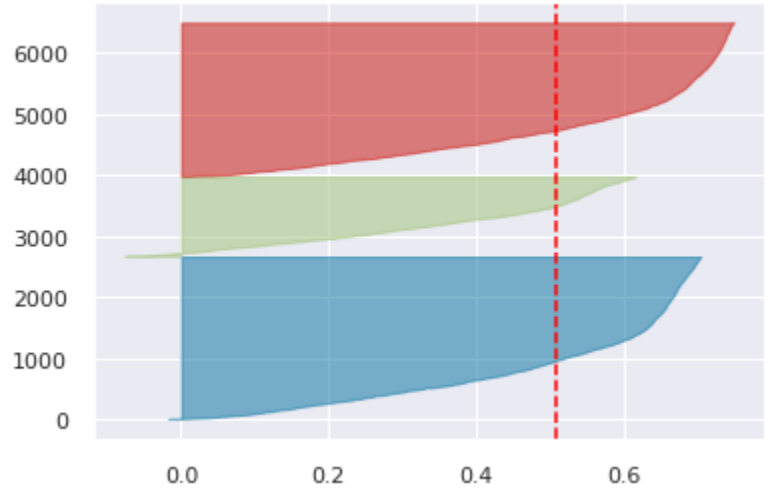

Elbow method
1e10
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
WCSS
1
2
3
4
5
6
7
8
9
Number of Clusters


K-Means Clustering
140000
120000
100000
80000
60000
WCSS Scores
2
4
6
8
10
Number of Clusters

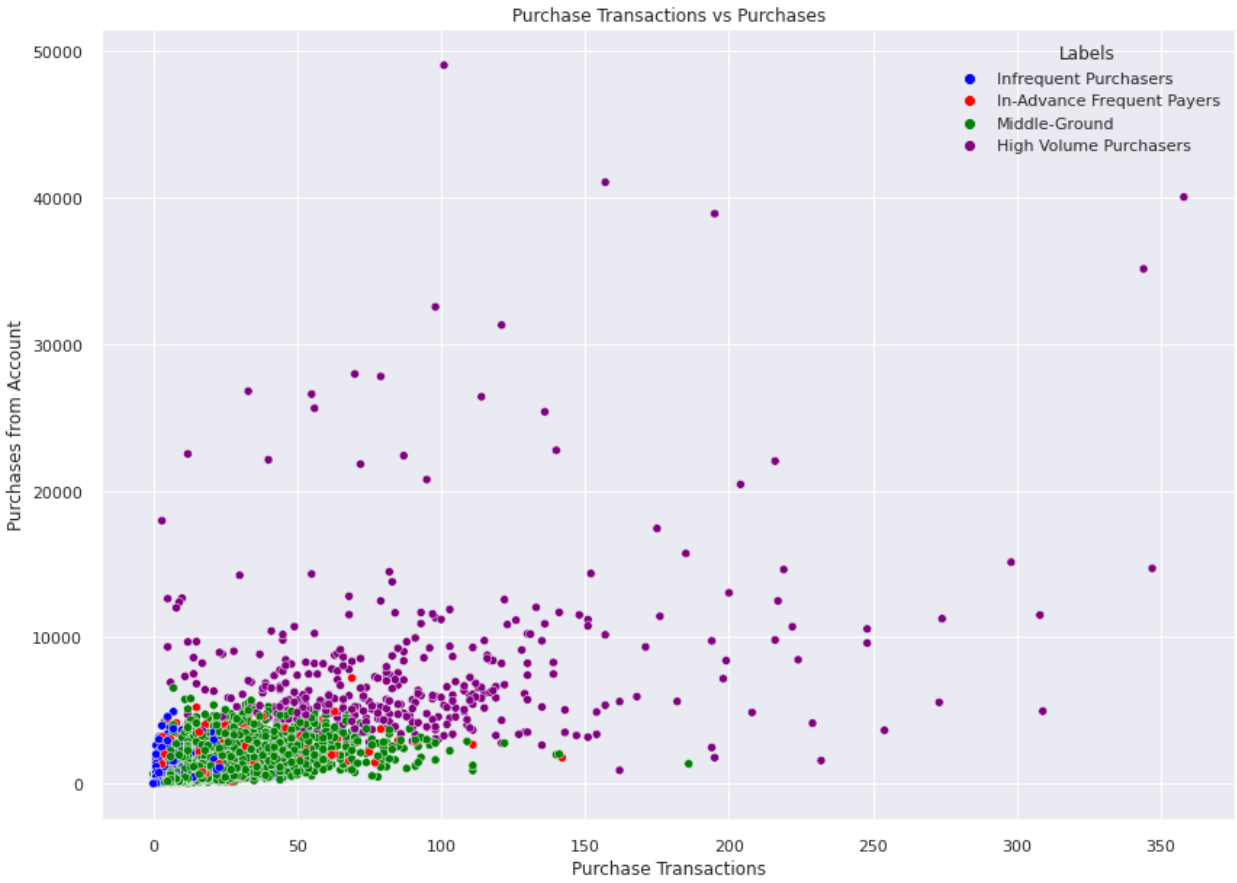
Бачите там лікоть? Здається, що лікоть становить близько 3 або 4. Ми будемо використовувати оцінку силуету, щоб побачити, який з них працює краще. Щоб прийняти рішення на основі дендрограми і нашої метрики суми квадратів всередині кластера, ми можемо запустити ще один kmeans з 4 кластерами і побудувати наш результат. Судити про це трохи складніше, оскільки у нас тут немає явного "ліктя".

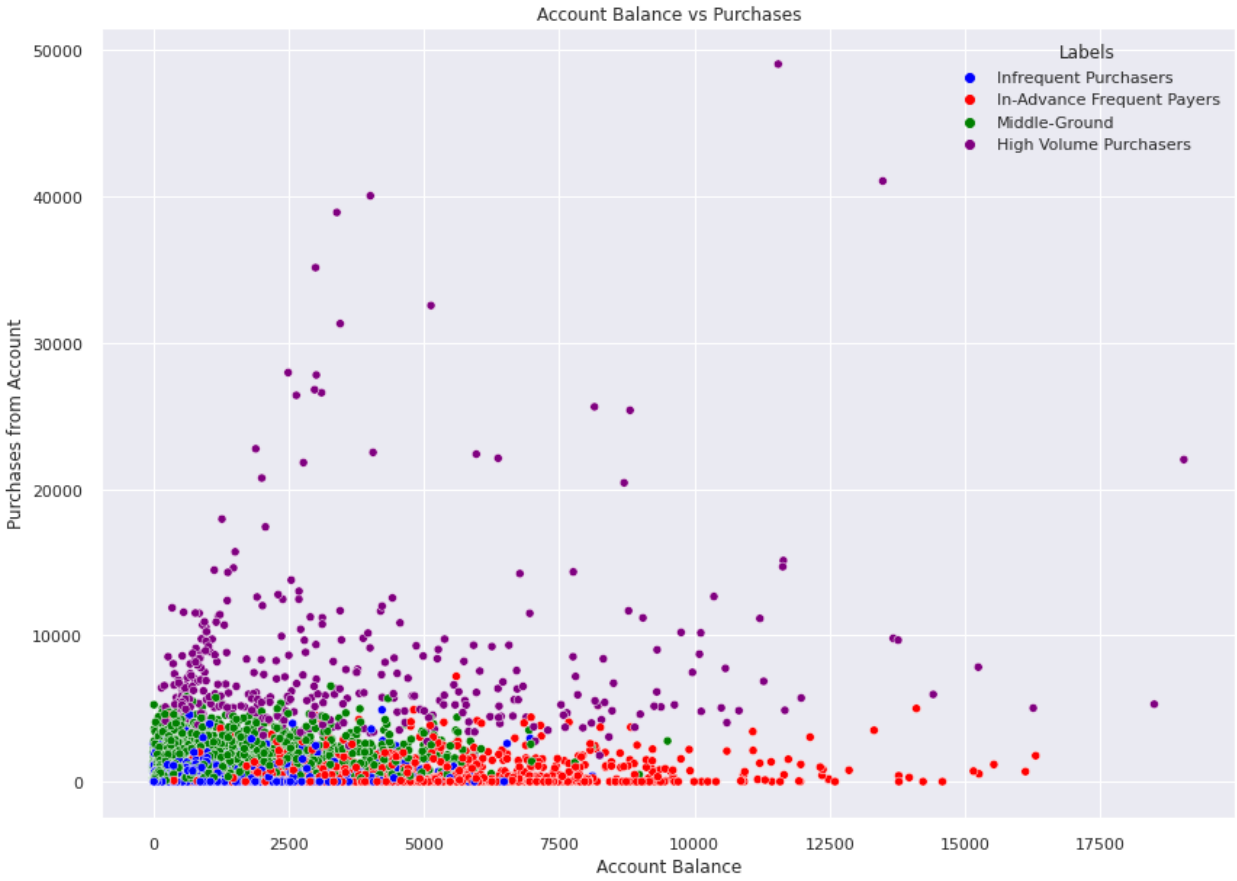


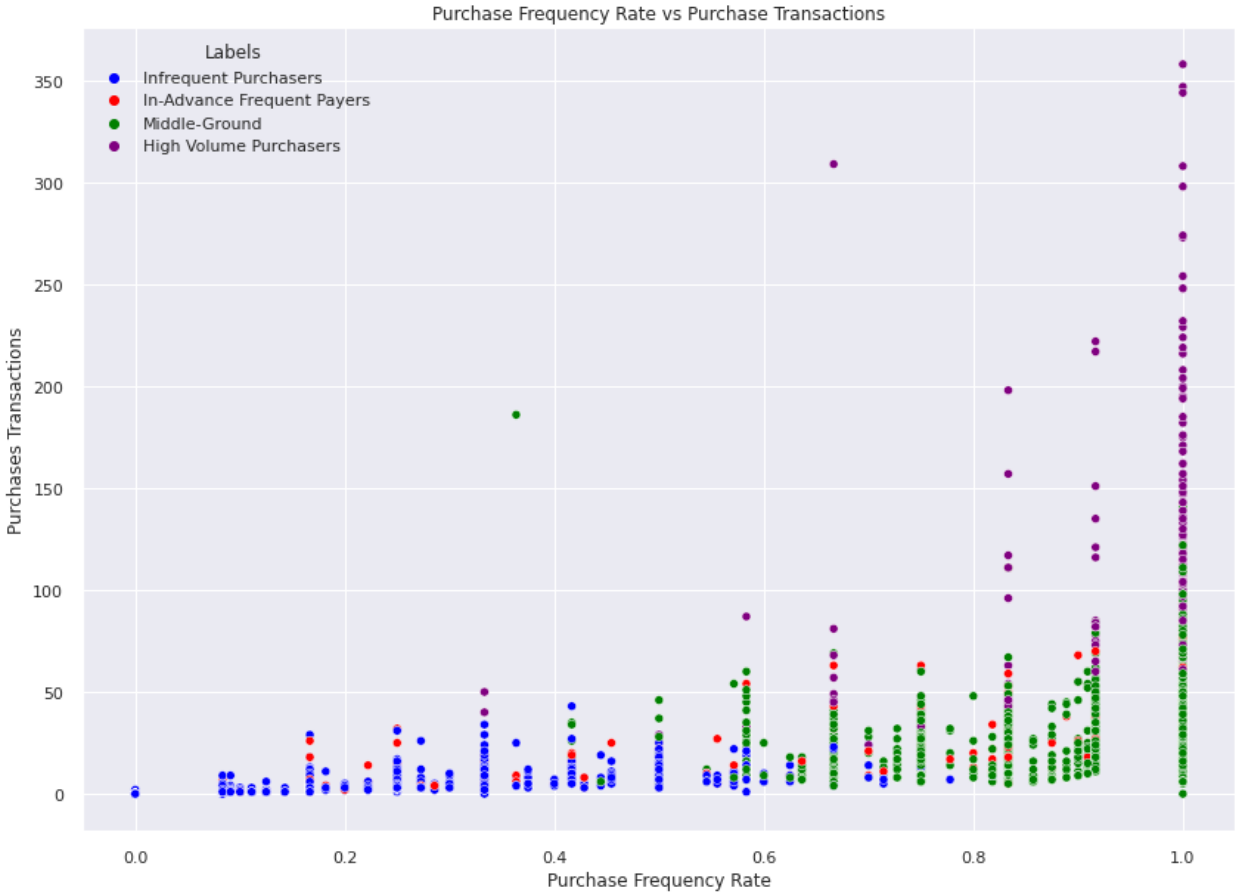
Добре, ми помилилися, n = 2, схоже, має більш високий силует, ніж два інших. Це те, що є, ми виберемо 2 Як нашу кількість кластерів.

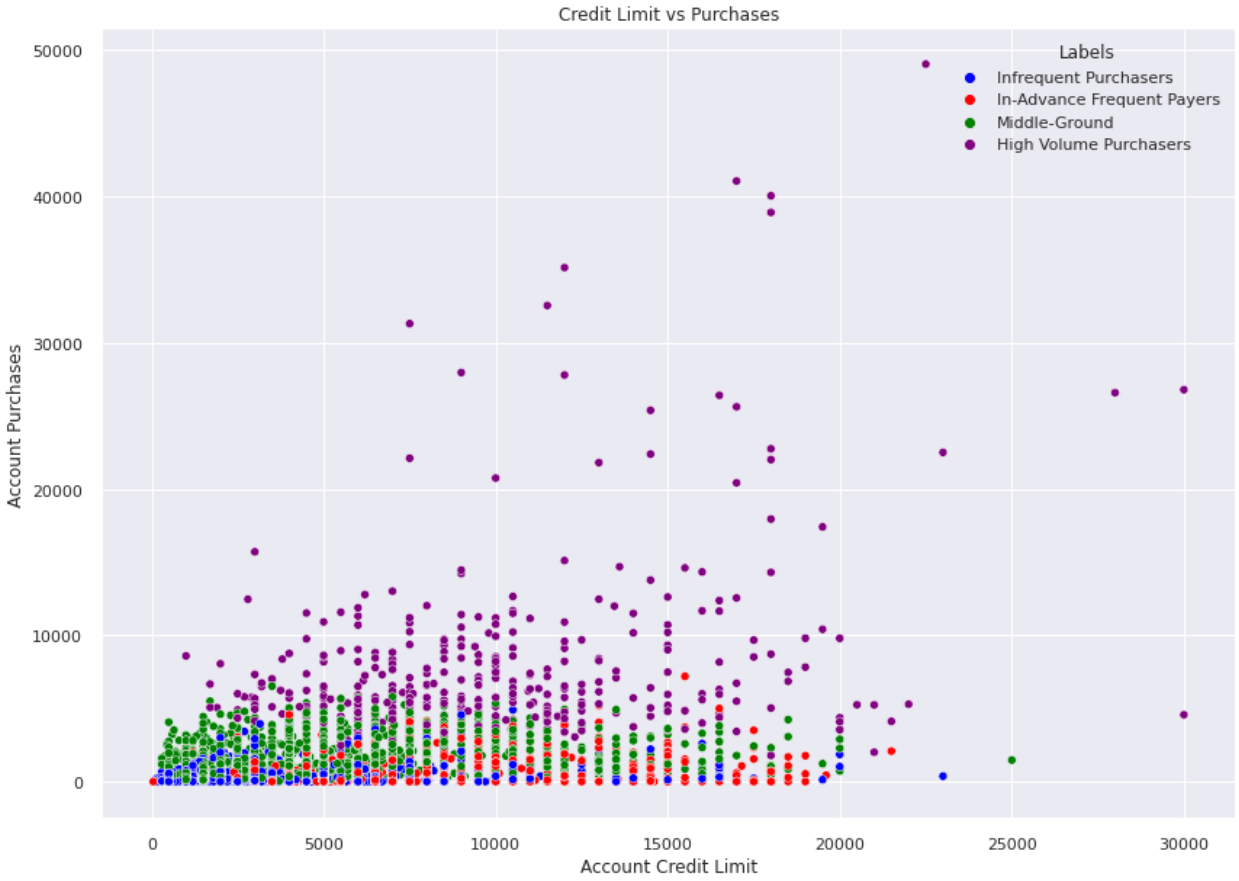


Аналіз сегмента K Means:



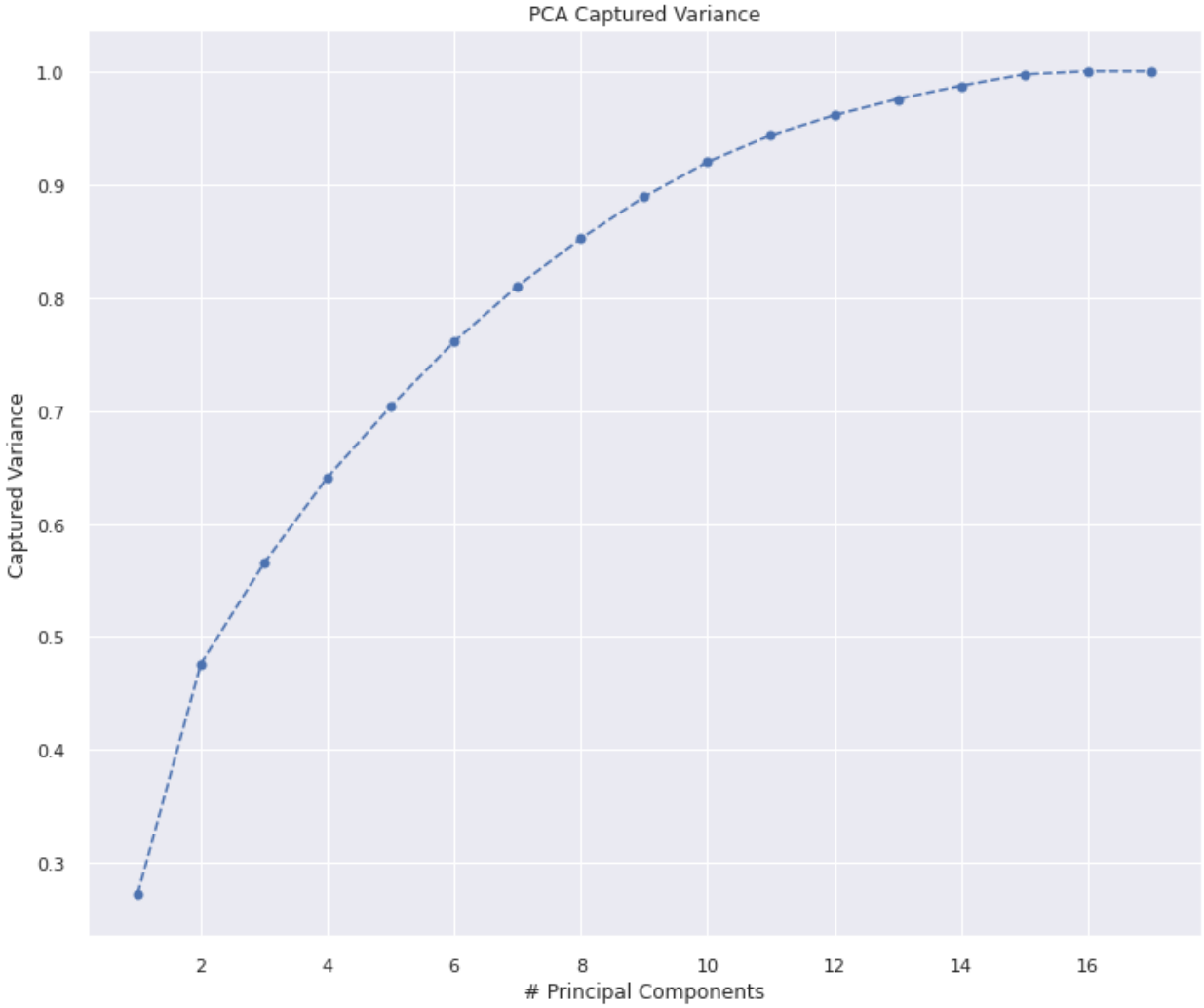

Account Balance vs Purchases
Labels
Infrequent Purchasers
In-Advance Frequent Payers
Middle-Ground
High Volume Purchasers
Purchases from Account
50000
40000
30000
20000
10000
0
0
2500
5000
7500
10000
12500
15000
17500
Account Balance


Purchase Frequency Rate vs Purchase Transactions
Labels
Infrequent Purchasers
In-Advance Frequent Payers
Middle-Ground
High Volume Purchasers
Purchases Transactions
Purchase Frequency Rate
0
50
100
150
200
250
300
350
0.0
0.2
0.4
0.6
0.8
1.0



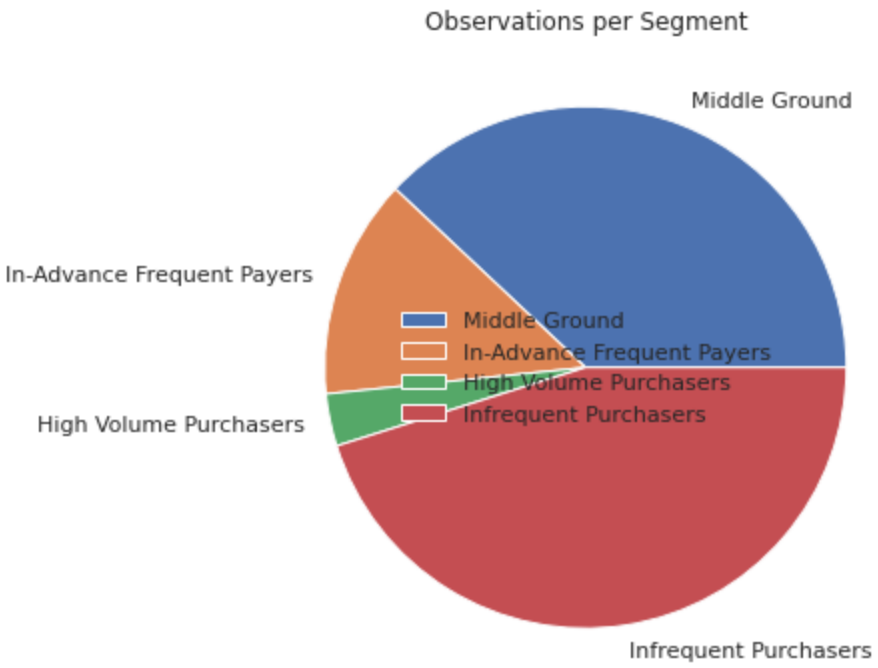
Дивлячись на наведені вище графіки, здається, що наша модель згуртувала клієнтів з низьким використанням кредитних карт в одному кластері і модель з більш високим використанням кластерів в іншому. Відмінно! ми відповідним чином розподіляємо ресурси. Особисто мені це здається кращою кластеризацією, оскільки вона дійсно сегментує верхню половину клієнтів, які використовують кредитні картки частіше, ніж зазвичай, і клієнтів з дуже низьким рівнем використання. Це здається більш дієвим результатом, якщо ми хочемо направити наші маркетингові стратегії відповідно до використання кредитної картки.

PCA



У нас є значна кількість компонентів, і, як і в нашому алгоритмі K Means, немає кристалізованого відсікання. Ми можемо спробувати встановити планку на рівні 80% дисперсії і зберегти 7 компонентів, щоб виключити їх з нашого аналізу тут і продовжити.

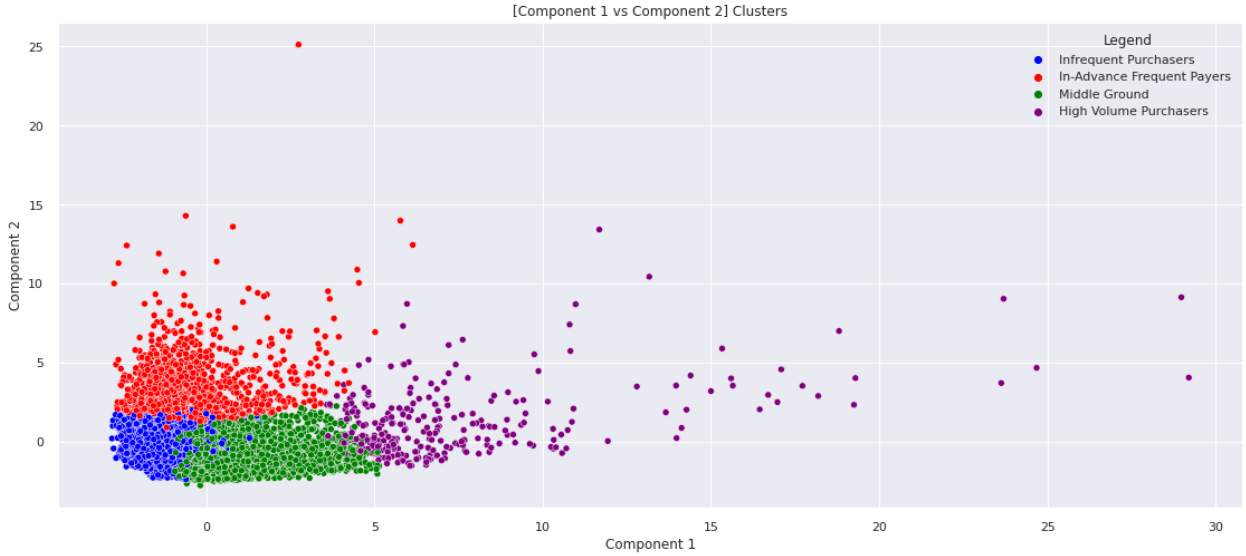
```
array([[ 0.0919859 ,  0.10981218,  0.41215123,  0.34677536,  0.33705564,
        -0.03058765,  0.32366488,  0.29476135,  0.27722626, -0.09914541,
        -0.05696036,  0.39106653,  0.21005184,  0.26372547,  0.05932632,
         0.13056503,  0.07791867],
       [ 0.4059787 ,  0.12773873,  0.0495303 ,  0.06992965, -0.01148132,
         0.43724688, -0.1865817 , -0.01474658, -0.17357691,  0.42999689,
         0.41641184, -0.0119466 ,  0.24382309,  0.26418176,  0.17041577,
        -0.1957089 , -0.00456558],
       [-0.17415522, -0.45885334,  0.24258187,  0.3685726 , -0.10375304,
        -0.00172594, -0.35574976,  0.10474308, -0.44994026, -0.08763546,
        -0.08705192, -0.07979903,  0.0951819 ,  0.28792071, -0.24870622,
         0.18419598, -0.06574319],
       [ 0.25942307,  0.15932011,  0.06400168,  0.12314791, -0.07502838,
        -0.26556462, -0.221738  ,  0.05546399, -0.26529256, -0.26659223,
        -0.33264408, -0.0241064 ,  0.12272574, -0.09751655,  0.35220392,
        -0.41815027,  0.42837395],
       [ 0.0757004 , -0.45085906, -0.01041   , -0.19702123,  0.33748918,
         0.09942509, -0.08853801, -0.5215596 ,  0.17540752, -0.15993226,
        -0.08974284, -0.05252272,  0.13200136,  0.18919193,  0.41681138,
         0.2010974 ,  0.11778693],
       [ 0.03576313, -0.01465339,  0.1959925 ,  0.17300614,  0.14543134,
        -0.13251141, -0.08569339, -0.09682787, -0.04745999,  0.03154756,
        -0.08978691,  0.07813781, -0.31286494, -0.06565158,  0.34027647,
        -0.28866061, -0.74566146],
       [-0.2633695 ,  0.09867483,  0.20135723,  0.11273384,  0.26897198,
        -0.03854   , -0.15790086, -0.30574575,  0.04322889,  0.13731595,
         0.19670125,  0.10423056, -0.54388385,  0.16879476, -0.20417945,
        -0.28035918,  0.40066207]])
```

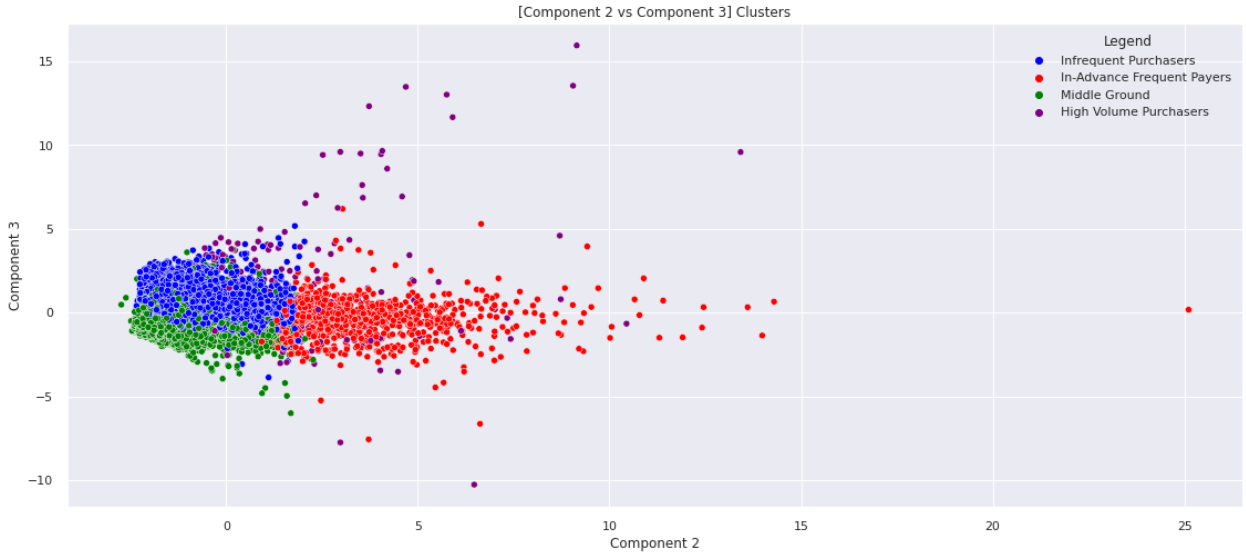


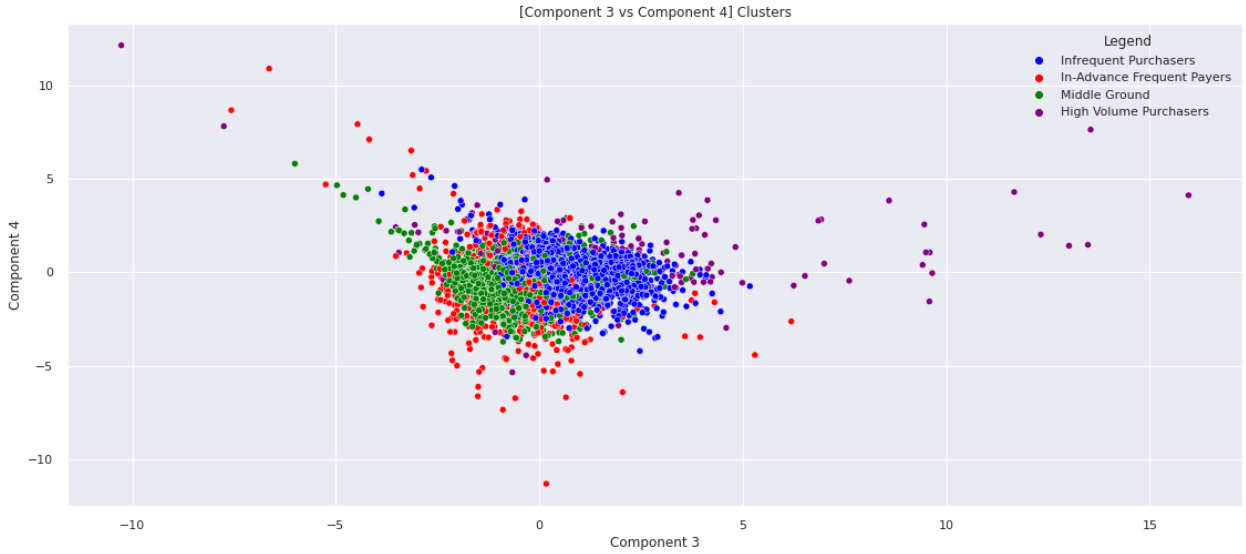
Порядок груп був змінений з тих пір, як ми включили компоненти PCA на основі нашої кластеризації k-середніх.

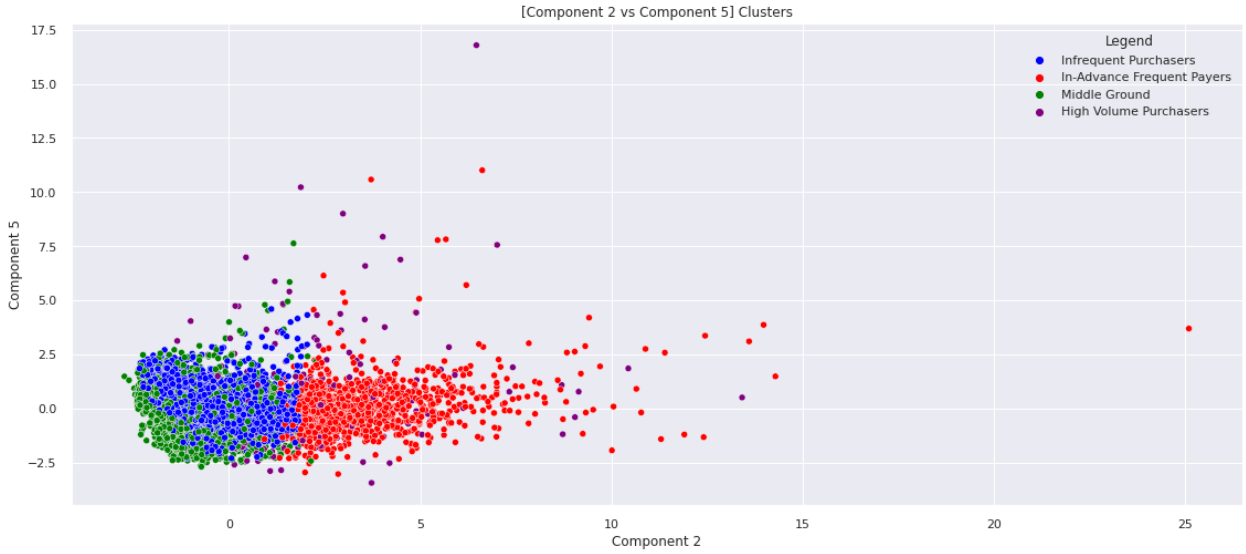
Давайте тепер нанесемо деякі компоненти PCA один на одного і розфарбуємо по наведеним нижче сегментам K Means. У нас є кілька компонентів PCA, тому було б трохи громіздко будувати графік і зберігати тут всі можливі комбінації компонентів.

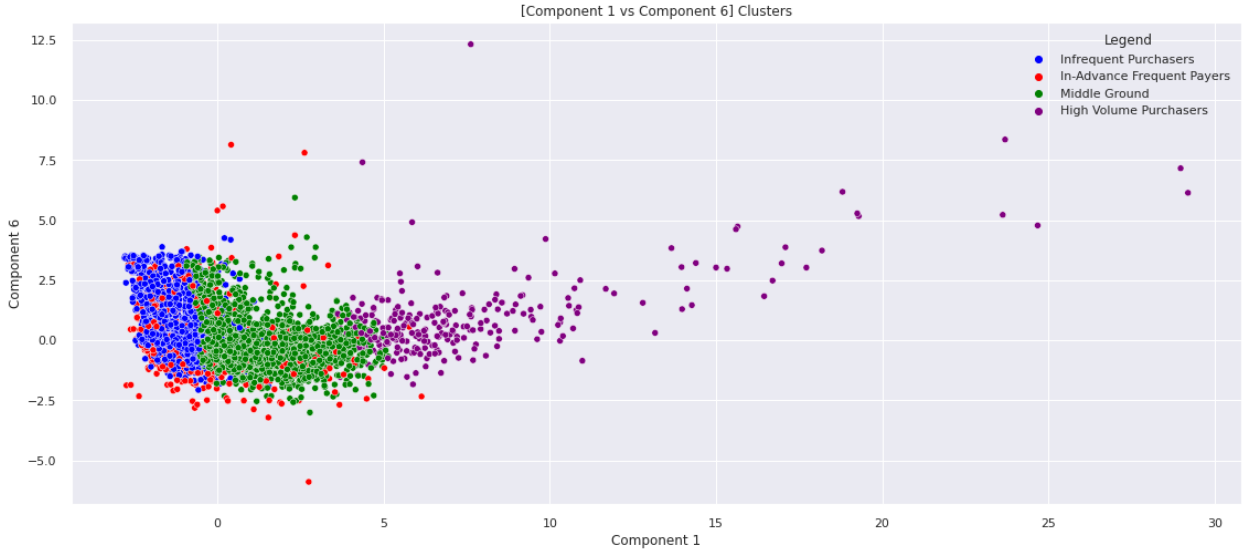
Отже, побудуємо графіків обраних компонентів.

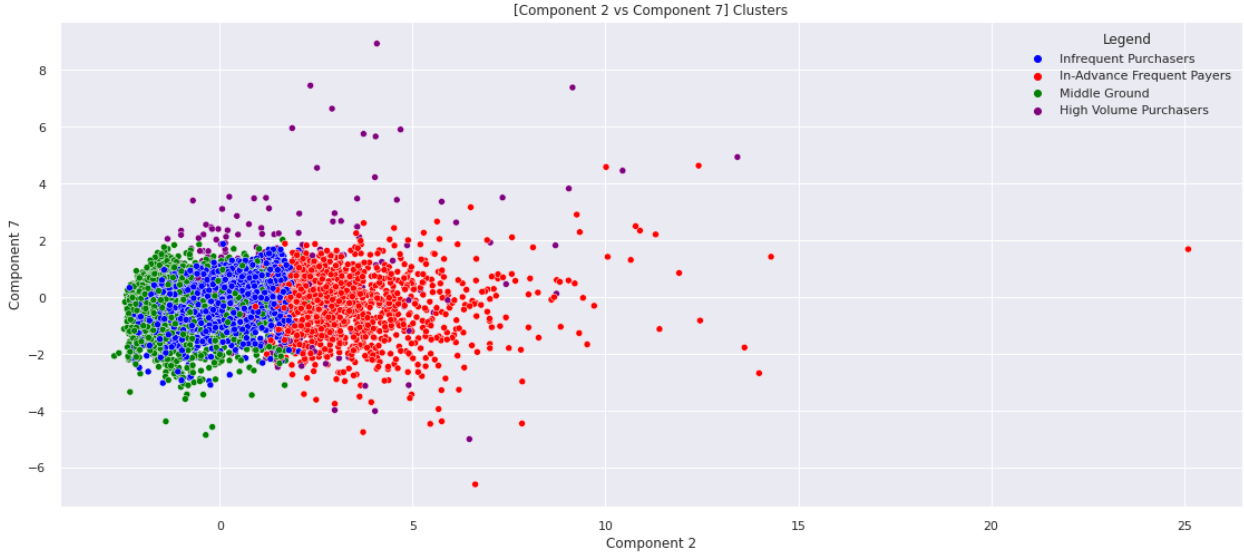

[Component 1 vs Component 2] Clusters
Legend
Infrequent Purchasers
In-Advance Frequent Payers
Middle Ground
High Volume Purchasers
Component 2
Component 1
25
20
15
10
5
0
0
5
10
15
20
25
30

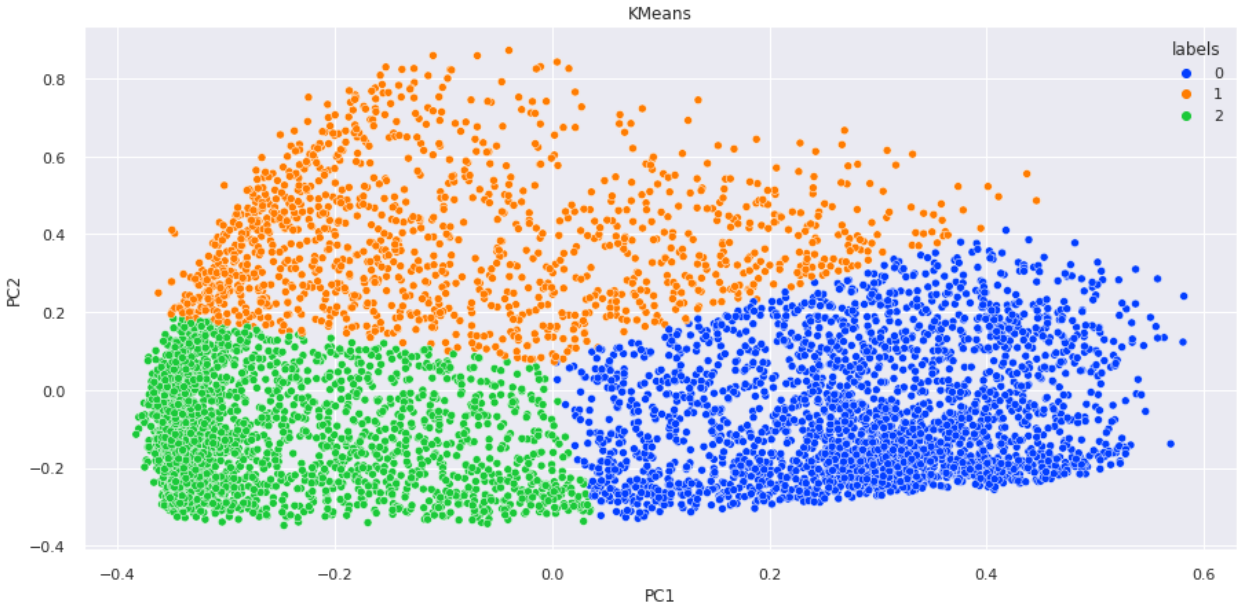

[Component 2 vs Component 3] Clusters
Legend
Infrequent Purchasers
In-Advance Frequent Payers
Middle Ground
High Volume Purchasers
Component 3
Component 2
15
10
5
0
−5
−10
0
5
10
15
20
25


[Component 3 vs Component 4] Clusters
Legend
Infrequent Purchasers
In-Advance Frequent Payers
Middle Ground
High Volume Purchasers
Component 4
Component 3
10
5
0
−5
−10
−10
−5
0
5
10
15


[Component 2 vs Component 5] Clusters
Legend
Infrequent Purchasers
In-Advance Frequent Payers
Middle Ground
High Volume Purchasers
Component 5
Component 2
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
0
5
10
15
20
25

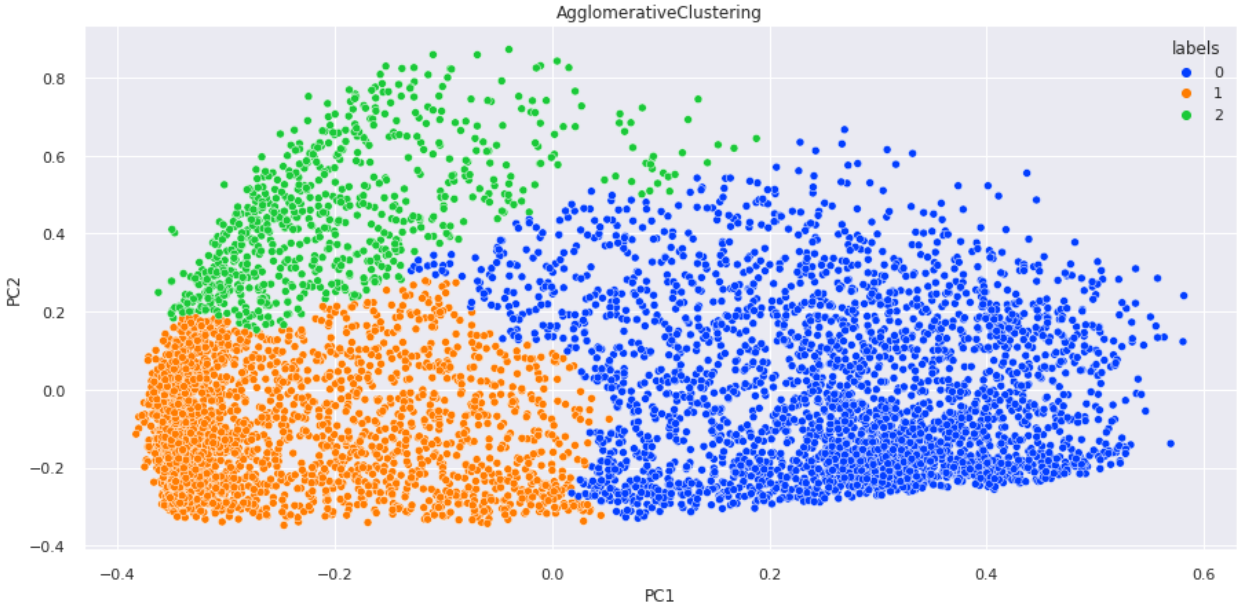

[Component 1 vs Component 6] Clusters
Legend
Infrequent Purchasers
In-Advance Frequent Payers
Middle Ground
High Volume Purchasers
Component 6
Component 1
12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
0
5
10
15
20
25
30


[Component 2 vs Component 7] Clusters
Legend
Infrequent Purchasers
In-Advance Frequent Payers
Middle Ground
High Volume Purchasers
Component 7
Component 2
8
6
4
2
0
−2
−4
−6
0
5
10
15
20
25



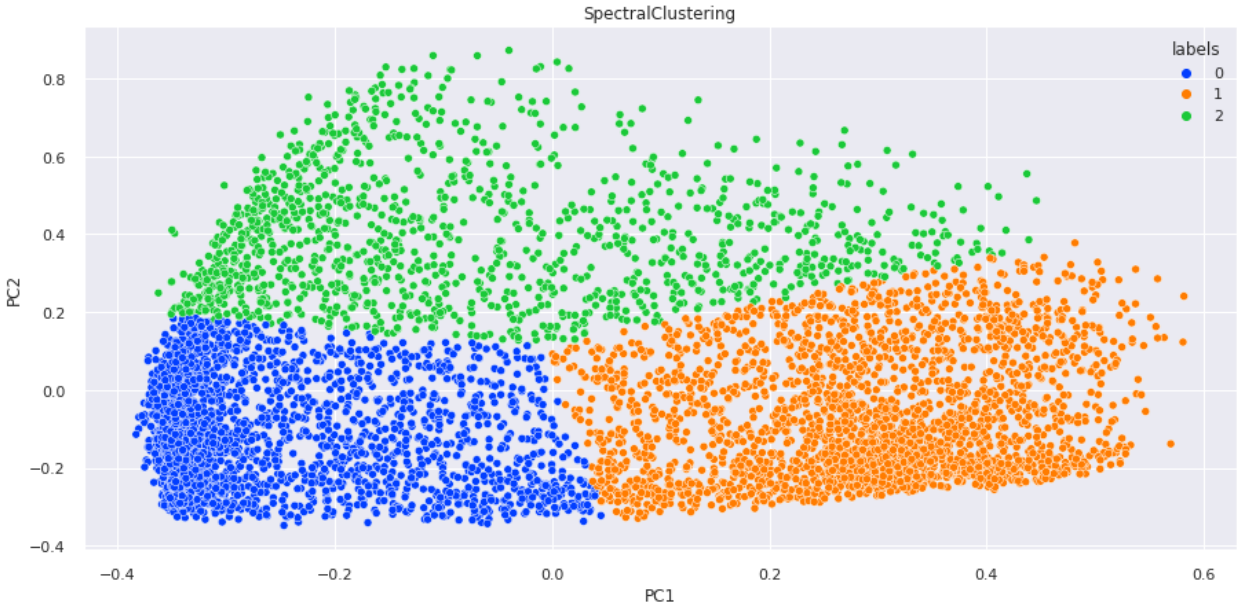
Silhouette Score : 0.51

Davies Bouldin Score : 0.71



Silhouette Score : 0.47

Davies Bouldin Score : 0.69



Silhouette Score : 0.51

Davies Bouldin Score : 0.71

Використані джерела

1. http://www.andriystav.cc.ua/Downloads/MITER/Lecture_07.pdf
2. https://russianblogs.com/article/2653561368/
3. https://github.com/antomys/ClusterizingData
4. Вэнь-Йен Чен, Янцю Сун, Хунцзе Бай, Чи-Джен Лин, Эдвард Я. Чанг. Параллельная спектральная кластеризация в распределенных системах.
5. https://code.google.com/p/pspectralclustering/
6. https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%BC_%D0%BA%E2%80%93%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BD%D1%96%D1%85
7. https://habr.com/ru/post/67078/