­

Geekbrains

**Анализ ассортимента он-лайн магазина косметики**

**на основе кластерного анализа покупательской оценки продукции**

Программа:

Искусственный интеллект.

Цифровые профессии.

Специализация:

Data Science

Дельмухаметова Регина Анваровна

г. Уфа

2024

**Содержание**

Введение

**Глава 1. Загрузка датасета и библиотек**

1.1. Библиотеки

1.2 Обзор датасета и описание признаков

1.3 Добавление признаков

1.4. Удаление столбцов

1.5. Работа с нулевыми значениями

**Глава 2. Предварительная обработка данных**

2.1 Анализ признаков. Визуализация

2.2 Корреляционный анализ

2.3 Изменение влияния признаков

2.4 Нормирование

2.5. Масштабирование

**Глава 3. Кластеризация**

3.1. Elbow метод

3.2 Модель Kmeans

3.3 Агломеративная кластеризация

3.4 Понижение признаков, метод РСА

3.5 Визуализация кластеров

**Глава 4. Анализ данных на основе кластеров**

4.1. Основные характеристики кластеров

4.2 Кластер

4.3 Анализ по предпочтениям

4.4 Анализ по категориям

4.5 Анализ по рейтингу свойств

Заключение

3 - 5

**3 - 10**

5 - 6

6 - 8

8

8 - 9

9 - 10

**10 - 20**

9 - 12

12 - 13

13

13 - 19

19 - 20

**20 - 24**

20 - 21

21

21 - 22

22 - 23

23 - 24

**24 -**

24 — 28

41 — 43

44

44 - 45

**Введение**

**Тема проекта:** Анализ ассортимента он-лайн магазина косметики на основе кластерного анализа покупательской оценки продукции и других параметров

**Цель проекта**: с помощью интрументов DataScience изучить данные сайта и проанализировать предпочтения по ассортименту на основе кластеризации характеристик о предпочтениях покупателей.

Представить современный магазин без сайта невозможно, и сейчас это не только инструмент для продаж, но и источник ценной информации. Как правильно использовать всю информацию о своих потребителях для улучшения ассортимента, сохранения и увеличения продаж. Один из подходов о удовлетворённости ассортиментом — это рейтинг продукции и другие действия посетителя сайта, которые подтверждают предпочтения того или иного ассортимента или качества.

В этом проекте есть готовый датасет по косметической продукции и всей информации с сайта магазина косметики. Датасет включает рейтинг товара, но **кластеризация** будет основываться не только на рейтинге, но и таких показателях, как:

- попадание или удаление из «листа желаний»,

- общем количестве отзывов,

- продукция, которую рекоментдуют или не рекомендуют к покупке и т. д.

На основе кластеров можно увидеть взаимосвязи между этими показателями и предпочтениями в свойствах продукции в разных категориях и назначениях.

**Проблема управления ассортиментом** обоснована всё большим количеством продукции и ограниченными ресурсами такими, как товарный запас, финансовые вложения, объёмы данных, людской ресурс для управления категорями, необходимостью понимания выбора именно «своего» клиента, чтобы сконцентрироваться на сильных сторонах предлагаемого ассортимента.

До обучения я работала более 20 лет в продажах FMSG и 11 лет в продажах кометики лидеров международных брендов (L’Oreal, Nivea, Vichy и т. д.). Моя работа заключалась в управлении и в переговорах с собственниками сетевых магазинов, где необходимы экспертность в категориальном менеджменте, маркетинге продвижения и понимания ключевых показателей бизнеса.

К off-line продажах крупных сетевых магазинах управление ассортиментом и выкладка товара выделена в отдельную дисциплину «катман», ктаегорийный менеджмент». Не редко услуги катмана передаются для управления крупным производителям, лидерам той или иной категории товара (например, Краски и Уход за волосами компании L’oreal, Гели для душа и Мужская серия для бритья компании Nivea).

On-line продажи растут быстрее и дают возможность получать обратную связь непосредственно от потребителя. Это преимущество можно использовать не только для управления своего ассортимента и взять как **инструмент изучения конкурентного on-line ритейла**. Кластеризация даст понять такие параметры, как:

* предполагаемый обём продаж, по количеству общих отзывов,
* ценовой сегмент в конкретной категории,
* важные свойства продукции для категории или ценового сегмента и т. д.

Задачи:

1. Скачать данные и зучить датасет
2. Изучить распределение признаков их влияние на кластеризацию
3. Изучить корреляцию между признаками. Удалить дублирующие признаки. Добавить новые характеристики.
4. Нормализовать и стандартизировать/масштабировать
5. Кластеризация
6. Проанализировать ассортимент по кластерам.

Посколько я только вхожу в область **Data Science** и AI, то выбрала тему для углубления в базовые инструменты анализа и как их можно использовать для лучшего управления ассортиментом. А именно **инструменты**:

- библиотеки pandas и numpy,

- язык программирования python, текстовый редактор markdown, и блокнот jupyter notebook,

- инструменты визуализации данных seaborn и matplotlib,

- инструменты для масштабирования и понижения признакового пространства от sklearn, MinMaxScaler и PCA,

- для кластеризации инструменты KMeans и AgglomerativeClustering и KelbowVisualizer от sklearn

**Глава 1. Загрузка датасета и библиотек**

**1.1. Библиотеки**

Буду использовать основные библиотеки для работы с табличными данными

* numpy — для использования математических и статистических функций с массивами данных
* pandas — для работы с большим табличными данные, БД, операциями над ними, а также инструмент статистического анализа и визуализации и т. д.
* jupyter notebook - для работы с табличными данными и текстовым редактором использую блокнот который позволяет работать с несколькими языками программированияб
* python версии 3.11 - для кода я использую,
* Markdown - текстовый редактор для текстового оформления и комментариев.
* matplotlib.pyplot и seaborn - для графической визуализации данных

От библиотеки машинного обучения Scikit-learn используем следующие инструменты:

* LabelEncoder — преобразования категориальных переменных в числовой формат,
* MinMaxScaler — масштабирование числовых признаков,
* PCA — principal component analysis, для снижения размерности данных,
* Kmeans — unsuprvised инструмент кластеризации,
* KelbowVisualizer — инструмент yellowbrick.cluster для визуализации «метода локтя»

Для улучшенря визуализации:

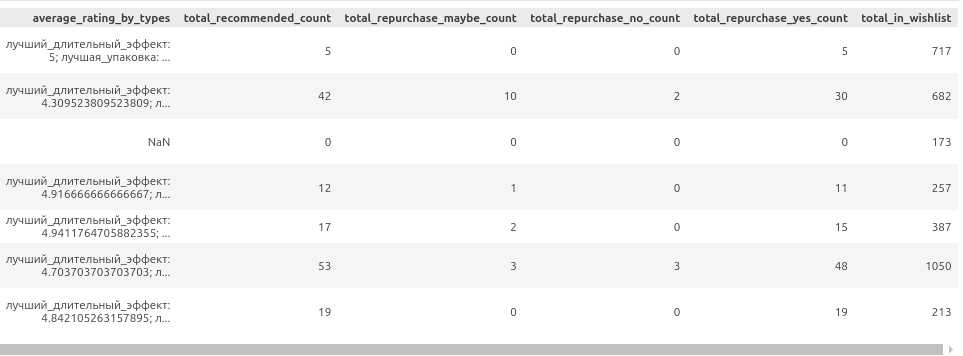
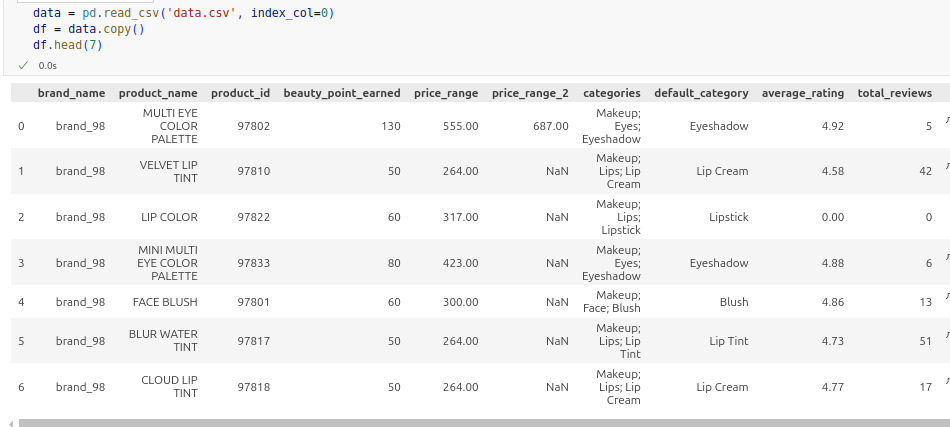
* %matplotlib inline
* %config InlineBackend.figure\_format = 'svg' — стиль для оформления графиков
* pd.options.display.max\_columns = 100 — улучшение видимости табличных данных (видимость максимально возможного кол-ва столбцов)
* pd.options.display.float\_format = '{:,.2f}'.format — ограничивать видимость чисел с «плавающей запятой»

**1.2 Обзор датасета и описание признаков**

Я использую готовые данные интернет-магазина косметики в формате scv, назовём датафрейм data. Все столбцы можно разделить на три блока:

* основные характеристики продукта (бренд, категория, ценовой диапозон)
* brand\_name - название бренда
* product\_id - id продукта
* product\_name - название продукта
* price\_range - нижняя граница цены продукта
* price\_range\_2 - верхняя граница цены продукта
* categories - категории
* default\_category — категория
* признаки, характеризующие выбор или предпочтения покупателя
* total\_reviews - количество отзывов
* total\_recommended\_count - количество рекомендаций
* total\_repurchase\_maybe\_count - количество возможного выкупа продукта
* total\_repurchase\_no\_count - количество без выкупа
* total\_repurchase\_yes\_count - количество выкупа
* total\_in\_wishlist - количество продукта в листе пожеланий
* признаки, с 5-ти бальной шкалой покупательской оценки оценки
* average\_rating - рейтинг по 5-ти бальной шкале
* average\_rating\_by\_types - отзывы с оценками

Посмотрим первые 7 строк:



Итого имеем:

- размер 7636 строк,

- 16 признаков,

- 5 строкового формата,

- 3 признака с неполными данными (price\_range\_2, categories, average\_rating\_by\_types)

Используем функцию для уменьшения объёма датасета, reduce\_mem\_usage. Она сокращает обём используемого памяти, за сч1т преобразования данных в соответствии с максимальными данными в признаке и строковый формат переводим в «category», который занимает меньший объём памяти. После преобразования сокращается объъём на 12,9%.

Посмотрим формат через атрибут dtypes

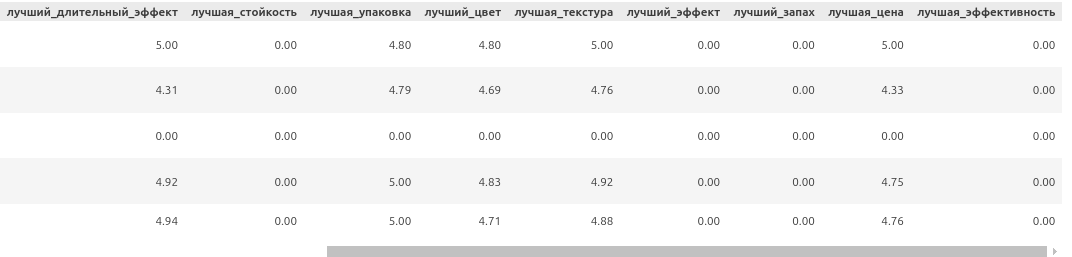
**1.3 Добавление признаков**

Первое, поскольку в признаке price\_range\_2 отсутсвует ~93% данных, то заменим два признака price\_range и price\_range\_2 на один price, как среднее значение между ними.

Второе, признак average\_rating\_by\_types оценки по свойтсвам продукта, мы разобьем его на отдельные столбцы с оценками в качестве значений.

(лучший\_длительный\_эффект: 5; лучшая\_упаковка: 4.8; лучший\_цвет: 4.8; лучшая\_текстура: 5; лучшая\_цена: 5)

Для этого перебёрем в цикле все строки исходного столбца, неизменяемый формат - множетсво , set(), позволит оставить только уникальные записи. Далее через цикл заполним значения по столбцам, разделяя каждую строку по знаку «;» и очищая от пробела, заполняем оценки в формате float. Получили 9 дополнительных признаков с оценками:



Разделение на отдельные столбцы даст нам более целевую кластеризацию, учитывая оцифрованный рейтинг от покупателя по каждому отдельному важному для потребителя свойству конкретного продукта.

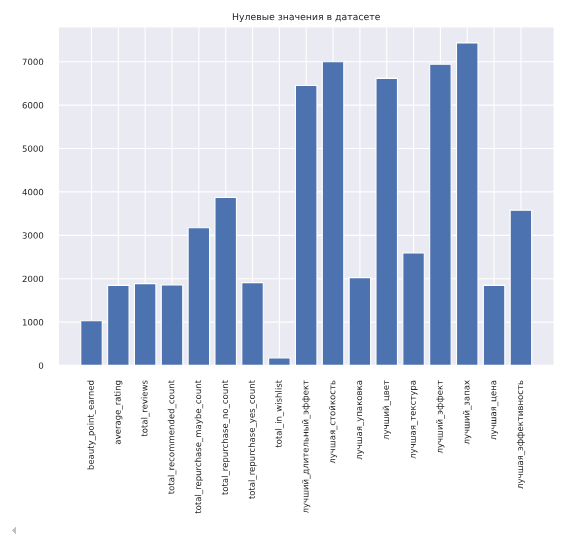
**1.4. Удаление столбцов**

Мы будем удалять те столбцы, которые не несут информацию о покупательском предпочтении:

* принадлежность к категории:
  + categories,
  + default\_categories,
* цена, price,
* product\_id, product\_name,
* average\_rating\_by\_types, котрый мы разделили на отдельные признаки

**1.5. Работа с нулевыми значениями**

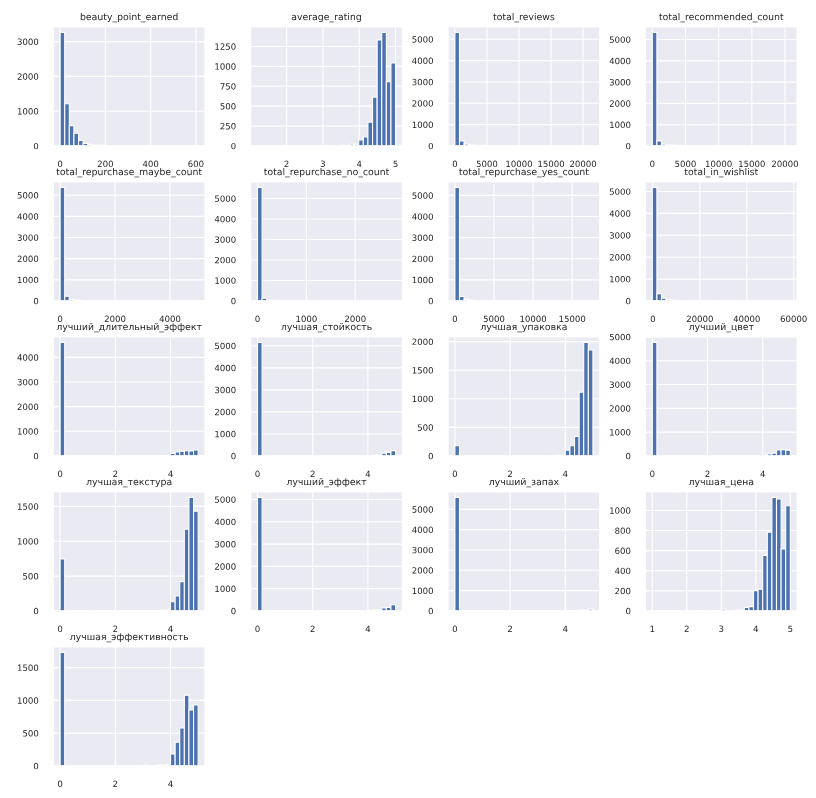
В этом пункте мы рассмотрим не пропуски, а именно нулевые значения. Для улучшения восприятия визуализируем по признакам и в соотношении с размером датасета в формате «bar», столбцов.



Добавленные нами признаки были образованны от столбца, пропуски в котором составляли 1842 строки. И как видно на графике, за исключением 2 признаков, остальные столбцы имеют значения равные 0 в этом же объёме, поэтому нам придётся сократить рамер датасета на 1842 строки.

**Глава 2. Предварительная обработка данных**

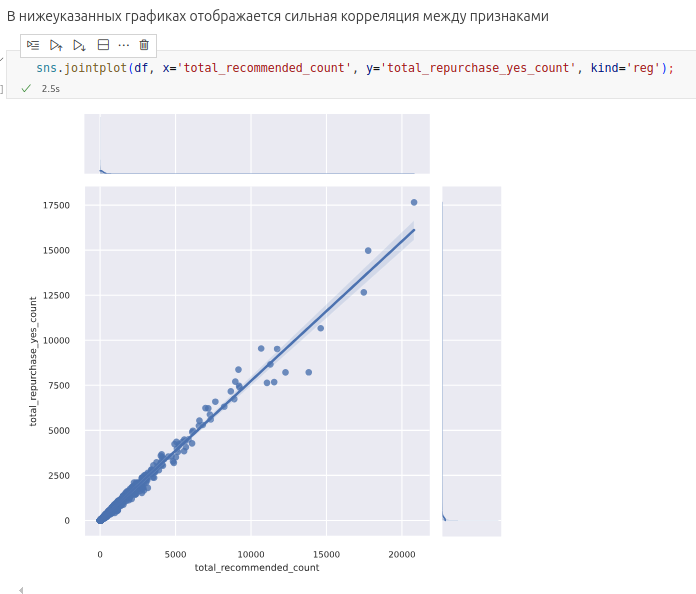
**2.1 Анализ признаков. Визуализация**

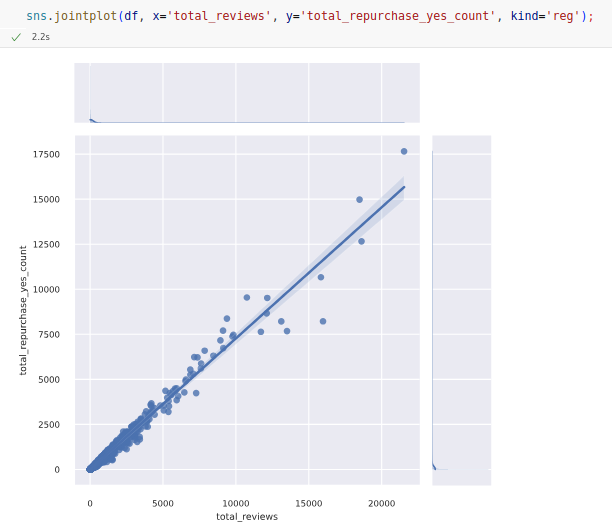
Выводы:

* все признаки имеют экспоненциальное распределение
* в характеристиках рейтинга подавляющее большинство значений лежит в промежутке от 4 до 5

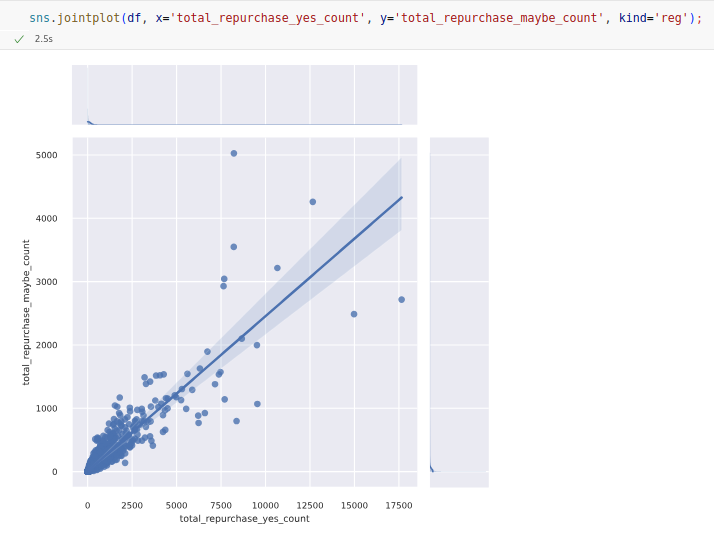
С помощью jointplot от seaborn посмотрим корреляцию между признаками

Видим три признака значительно коррелированных между собой





Эти два признака противоположны по значению



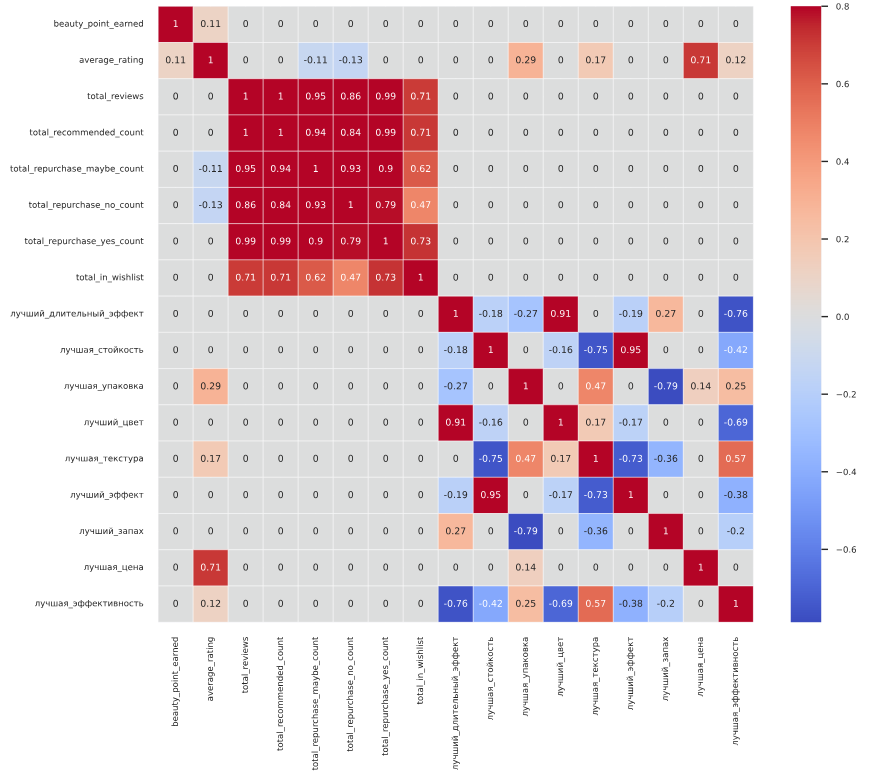
Признак `total\_repurchase\_no\_count` должен находится в обратной зависимости, а значит мы изменим значение на 1/признак

**2.2 Корреляционный анализ**

Для отображения всех признаков на предмет взаимосвязи между собой удобнее использовать «тепловую карту» или **heatmap** от seaborn. Корреляционный датасет получим с помощью функции от pandas — **corr()**, которая по умолчанию использует стандартный коэффициент корреляции.

Удалим сильно коррелированные признаки:

* total\_reviews,
* total\_recommended\_count,
* total\_repurchase\_maybe\_count

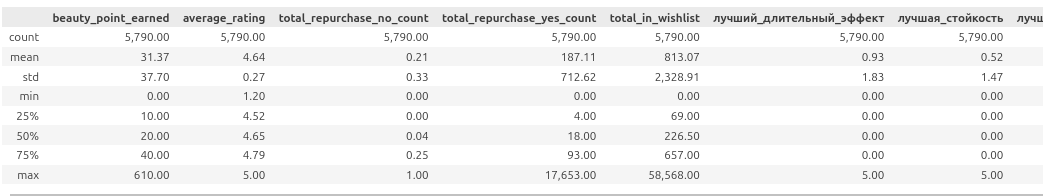


**2.3 Изменение влияния признаков**

В датасете присутствуют отрицательные значения в признаке «total\_in\_wishlist», т. е. продукт удалили из «избранного». Эти значения мы также поставим в обратную зависимость и изменим на 1/признак.

**2.4 Нормирование**

Метод describe() от pandas позволяет увидеть статистики по признакам. Как видно, при экспоненциальном распределении большая дисперсия, данные очень неравномерные и имеют много выбросов.

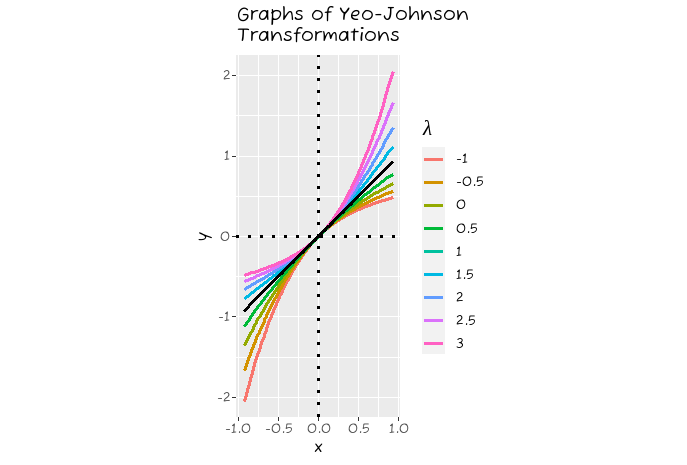


В этом случае данные необходимо нормализовать, для уменьшения влияния выбросов. существует несколько вариантов нормализации данных. Наиболее распространённые среди них:

* ****Логарифмическая трансформация** -** один из наиболее распространенных методов нормализации данных с экспоненциальным распределением. Логарифмическая трансформация помогает уменьшить асимметрию данных, делая их более симметричными и ближе к нормальному распределению. Функция **np.log()** из библиотеки numpy вычисляет натуральный логарифм (логарифм по основанию e, где e ~2,718)
* **Корневая трансформация** (square root), также помогает уменьшить асимметрию распределения данных, особенно если логарифмическая трансформация оказывается слишком сильной. Также доступна в библиотеке numpy, np.sqrt()
* ****Box-Cox transformation.**** Трансформация Бокса-Кокса является более универсальным методом, который может привести данные к нормальному распределению. Она использует параметр λ, который выбирается таким образом, чтобы сделать данные как можно более нормальными. Функция трансформации Box-Cox есть в библиотеке scipy.stats
* **Yeo-Johnson трансформация**. Это вариант трансформации **Box-Cox**, который может быть применен и к данным, включающим отрицательные значения.

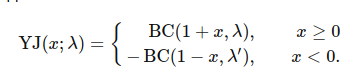
Методы логарифмической трансформации и Box-Cox применяются к положительным данным, поскольку если элемент массива меньше или равен нулю, то логарифм не определен, и функция выдаст предупреждение или ошибку (например, -inf для нуля и nan для отрицательных чисел). Поэтому я добавляю произвольное минимальное число к значениям для сдвига от 0.1 Отрицательных значений в датасете нет.

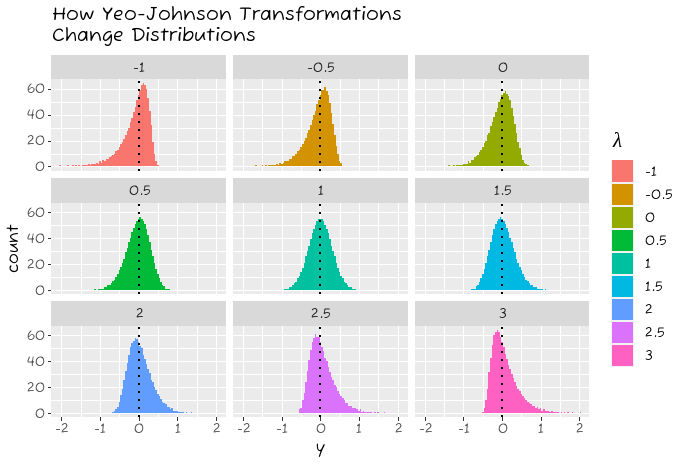
Как уже указывалось выше преобразование Yeo-Johnson применимо ко всем числам. Это делается путем разделения реальной линии на ноль, сдвигая положительные значения на 1 и отрицательные значения по − 1, применяя преобразование **Box-Cox** к абсолютным значениям, и отрицая их, когда аргумент отрицательный. По сути, он объединяет два преобразования **Box-Cox**. Однако они имеют «обратные» параметры **Box-Cox**. Естественным происхождением параметров, котрого является λ=1 и «обратный» параметр

λ′=2−λ, отражая линию параметра вокругλ=1.

Сдвиг происходит гладко, поскольку все преобразования Box-Cox задуманы так , чтобы согласовываться с тождественным преобразованием в точкеx=1.

Преобразование Yeo-Johnson

аглядно работу трансформации можно увидеть на нормальном распределении на изображении ниже

Теперь посмотрим, как эти преобразования отразятся на наших данных. Для трансформации сразу по всем видам создадам функцию data\_normalizer. Наиболее быстрый метод оценки визуализация данных на boxplot или ящике с усами от seaborn. Сравним графики по каждому виду трансформации с исходными данными, представленные ниже в следующем порядке:

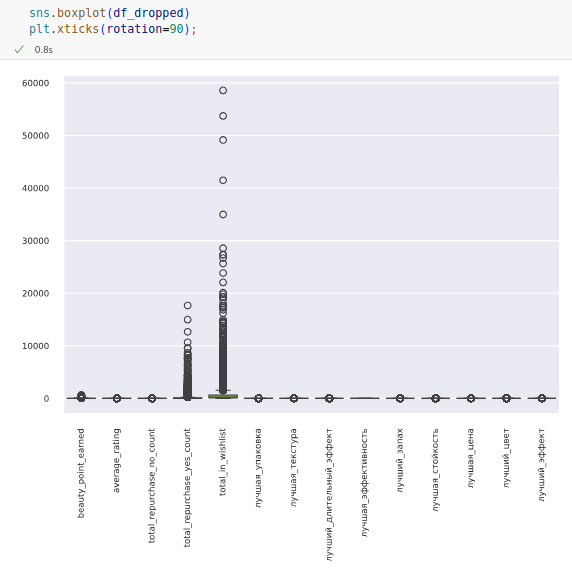
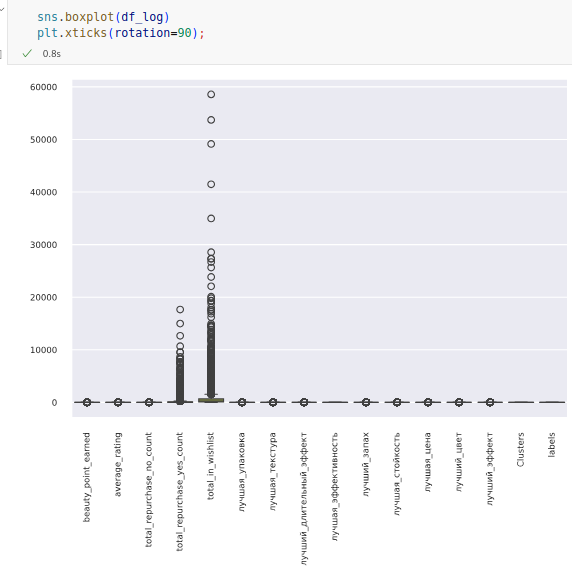
- исходный датасет,

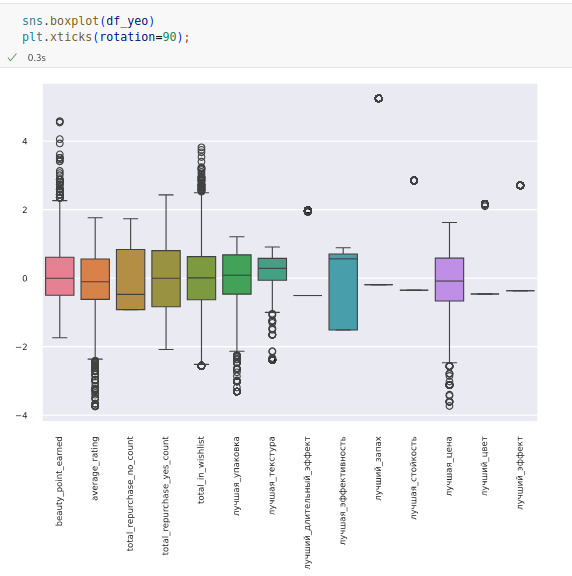
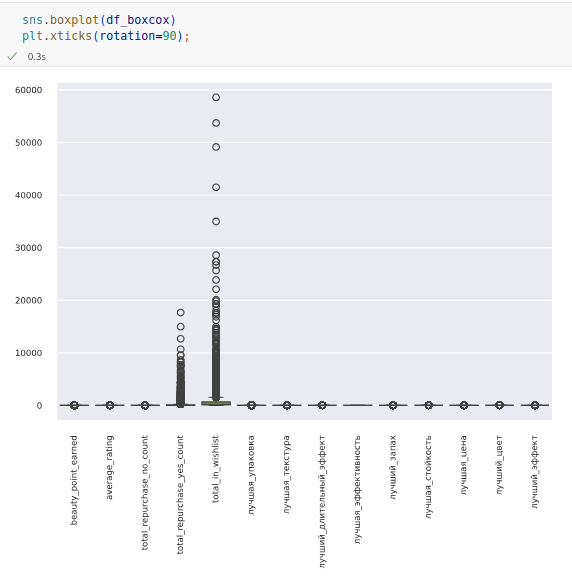
- логарифмическая трансформация,

- Корневая трансформаци,

- Boxplot,

- Yeo-Johnson





Как можно убедиться метод трансформации Yeo-Johnson дал наилучшие результаты, график распределения данных по признакам ниже, подтверждает стремление у нормальному распределению

**2.5. Масштабирование**

Далее можно масштабировать данные, используя функцию ***MinMaxScaler*** от sklearn.preprocessing, который принимает минимальное и макимальное значение, создадим экземпляр класса.

MinMaxScaler стандартизирует признаки в заданном диапазоне, в нашем случае: (0, 5).

X\_std = (X - X.min(ось=0)) / (X.max(ось=0) - X.min(ось=0)) X\_scaled =

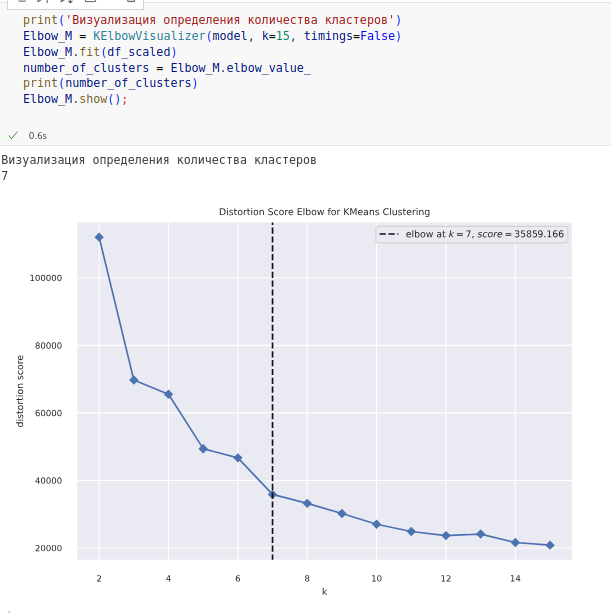
X\_std \* (макс - мин) + мин

Далее вызвав метод fit\_transform, передадим нормализованный датафрейм, заполним полученными данными новый датафрейм ***df\_scaled.***

**Глава 3. Кластеризация**

**3.1. Elbow метод**

Метод локтяВ KElbowVisualizerнем реализован метод «локтя», который поможет выбрать оптимальное количество кластеров, используя в модели диапазоном значений. Точка перегиба кривой является показателем того, что базовая модель лучше всего подходит для этой точки. В визуализаторе «локоть» отмечено пунктирной линией.



Переменной number\_of\_cluster обозначим оптимальное количество кластеров из данных модели.

**3.2 Модель Kmeans**

Задача кластеризации - разбить объекты из выборки на определённое число групп (или кластеров) так, чтобы "похожие" объекты попали в один кластер.

Я использую два из наиболее известных алгоритмов кластеризации - k-means или k средних и агломеративную кластеризацию. Данная модель принимает на вход параметр n\_clusters, т.е. число кластеров, на которые мы хотели бы разбить наши данные.

На начальной итерации алгоритм произвольным образом выбирает центры кластеров - центроиды (столько, сколько мы задали в параметре n\_clusters). Затем повторяется следующий процесс. Отнести каждый объект из выборки к тому кластеру, чей центр ближе всего к этому объекту. Переставить центроид каждого кластера в центр этого кластера (с учётом новых объектов). Этот процесс повторяется до тех пор, пока координаты центров не перестанут изменяться.

В итогея получаю 7 кластеров и суммарным кол-вом данных по каждому кластеру

Clusters

4 1585

0 1284

5 1143

3 704

1 595

6 316

2 163

**3.3 Агломеративная кластеризация**

Перейдём к иерархической кластеризации, которая в свою очередь состоит из агломеративной и дивизивной кластеризации.

Алгоритмы агломеративной кластеризации основаны на объединении мелких кластеров в более крупные. На начальной итерации каждый объект считается отдельным кластером, затем происходит их последовательное объединение.

Напротив, дивизивная кластеризация - это обратный процесс: сначала вся выборка считается одним большим кластером, затем происходит последовательное разбиение кластеров на части.

Кластеры по количеству данных незначительгно отличаются между собой.

labels

3 1489

1 1315

5 1125

0 1020

2 550

4 164

6 127

**3.4 Понижение признаков, метод РСА**

Метод главных компонент (PCA, Principal Component Analysis) — это широко используемая техника снижения размерности, которая применяется для анализа данных и уменьшения их сложности, сохраняя при этом максимально возможную часть информации.

Цель PCA заключается в преобразовании исходного набора переменных в новый набор переменных, называемых главными компонентами. Эти главные компоненты — линейные комбинации исходных переменных, упорядоченные по убыванию дисперсии. То есть, первая главная компонента (PC1) объясняет максимальную дисперсию в данных, вторая компонента (PC2) объясняет следующую по величине дисперсию, и так далее.

Основные шаги метода PCA

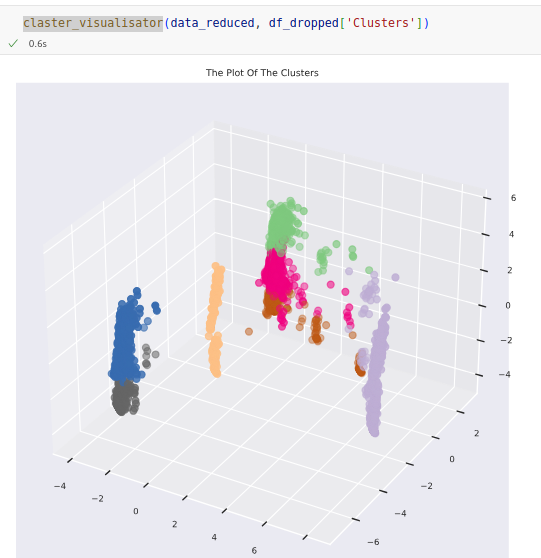
* Центрирование данных . Все переменные в данных центрируются, т.е из каждого значения переменной вычитается её среднее значение, чтобы среднее каждой переменной стало равным нулю.
* Выборка ковариационной матрицы. Строится ковариационная матрица для центрированных данных. Эта матрица показывает, как переменные в данных изменяются относительно друг друга.
* Нахождение собственных значений и собственных векторов. Собственные значения и собственные векторы ковариационной матрицы вычисляются. Собственные значения показывают, сколько дисперсии объясняет каждая из главных компонент, а собственные векторы определяют направление главных компонент в пространстве данных.
* Построение главных компонент . Главные компоненты формируются как линейные комбинации исходных переменных с коэффициентами, равными соответствующим собственным векторам. Полученные компоненты упорядочиваются по убыванию собственных значений.
* Выбор числа компонент. Часто выбирается подмножество главных компонент, которое объясняет достаточную долю общей дисперсии данных (например, 95% или 99%).
* Преобразование данных. Данные проецируются на новое пространство, определяемое выбранными главными компонентами, что приводит к их снижению размерности.

Для PCA метода создам функцию feat\_reducer, которая ринимает на вход датасет и кол-во компонент.

**3.5 Визуализация кластеров**

Для визуализации в 3d будем использовать 3 размерности — 3 компоненты

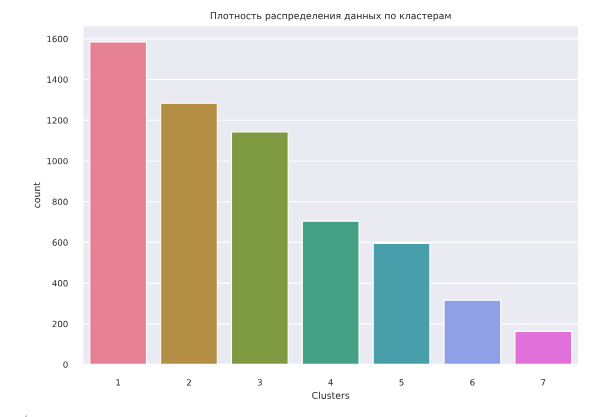
А также создам функцию claster\_visualisator для графика scatter от matplotlib.pyplot



**Глава 4. Анализ данных на основе кластеров**

**4.1. Основные характеристики кластеров**

Для улучшения восприятия информации я поменяла номерацию кластеров в соответствии с распределением данных. С помощью seaborn.countplot посмотрим плотность распределения данных по кластерам.

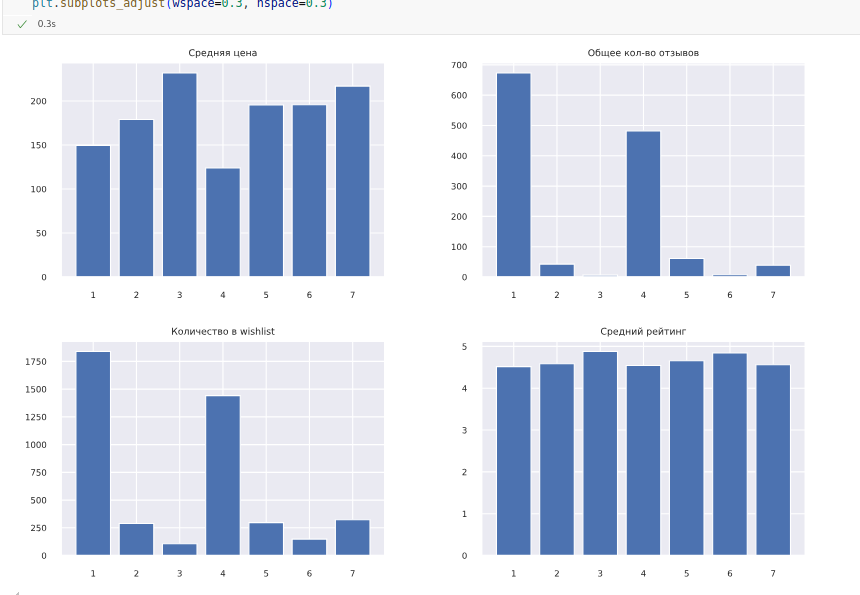
.

Посмотрим какие характеристики присущи каждому кластеру. Для этого мы будем использовать копию изначально загруженного датасета, проведя необходимые преобразования через созданную функцию **data\_transformer**:

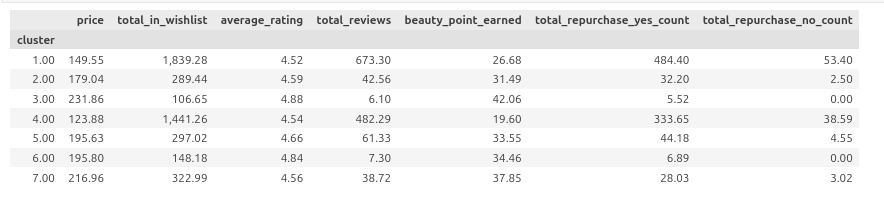
* создадим копию датасета с помощью pandas.copy(),
* заменим столбцы по ценам на один,
* удалим столбцы, которые не несут информации для анализа, либо дублируются

На графиках по основным характеристикам продукта видно различия кластеров:

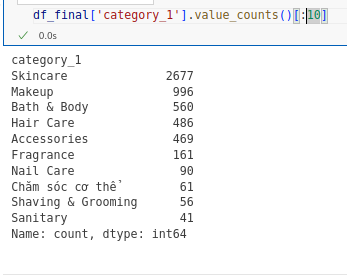
* Для **кластеров** с минимальными ценами **1** и **4** характерна массовость , а значит больше отзывов и больше добавление в лист\_желаний.
* И обратная ситуация с **кластером 3** с максимальным уровнем цен, но у этого кластера самый высокий средневзвешанный рейтинг.

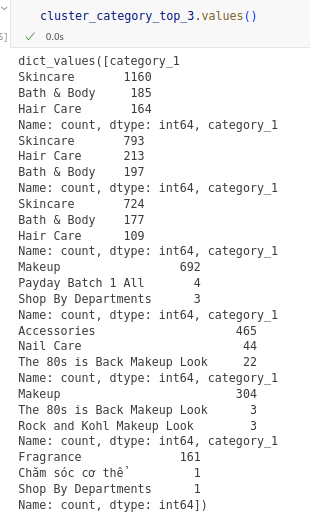


* **Кластер 2** и **5** отличает от остальных в признаках по выкупу и возврату продукта, total\_repuchase\_yes\_count и total\_repuchase\_no\_count



Top — 10 по категорий:

Top 3 категори в каждом кластере

Кластер 1

Кластер2

Кластер3

Кластер4

Кластер 5

Кластер 6

Кластер 7

Заключение

Вывод:

**в кластере 7** — занимает топ рейтинг по эффекту «лучший запах», также он в лидерах по уровню цен, а можно сделать вывод, что покупатель предпочитает лучший запах цене.

В **кластере 1** и 3 — основные по категории Skincare, но при этом оценка эффективности в кластере 3 выше (4,47 и 4,87 соответственно), там где выше цена, что говорит о в средствах по уходу за лицом покупатель предпочтёт эффективность цене

**Кластер 6**, в котором присутсвует одна категория Makeup (макияж) является лидерам по таким эффектам, как:

* лучший\_длительный\_эффект
* лучший\_цвет
* лучшая\_упаковка
* лучшая\_цена

Т.е. те эффекты, по которым покупатель делает выбор, но он платит средняя цену, а значит готов платить за выше указанные эфффекты.

**Заключение**

В данном примере я намеревалась показать как на соновании характеристик можно понять предпочтения покупателя, основные предпочтения и какую цену он готов платить.

Очень показательным считаю те инструменты, которые качественно сформировали клстеры, а именно:

* нормализация Yeo-Johnson,
* Кластеризация KNN
* инструменты визуализации обелгачают сортировку признаков по корреляции и ускоряют понимание данных при работе с датасетом

­

К сожалению, я не успела глубоко проанализировать продуктовые линейки, но уже этот анализ позволяет понять, как кластеризация значительно помогает ориентироваться в ассортименте и его покупательских предпостениях.

Благодарю за внимание.