Результаты сопоставления элементов A и B

Мастерская 2

Постановка задачи

Задача:

Даны наборы данных A и B. Необходимо придумать алгоритм сопоставления элементов B элементам A при условии, что часть элементов B сопоставлены элементам A.

Метрика качества:

$$\text{accuracy@5} = \frac{\sum[\text{среди 5 найденных элементов есть нужный}]}{\text{количество сопоставляемых элементов}}$$

Данные:

- Полный набор данных: |A| ~ 3 млн. |B| ~ 200 тыс.
- Сокращенный набор данных: |A| ~ 300 тыс. |B| ~ 20 тыс.

Метод и этапы решения

Метод решения:

Для сопоставления элементов A и B достаточно найти метрику расстояния ρ(a, b) характеризующую их близость между собой.

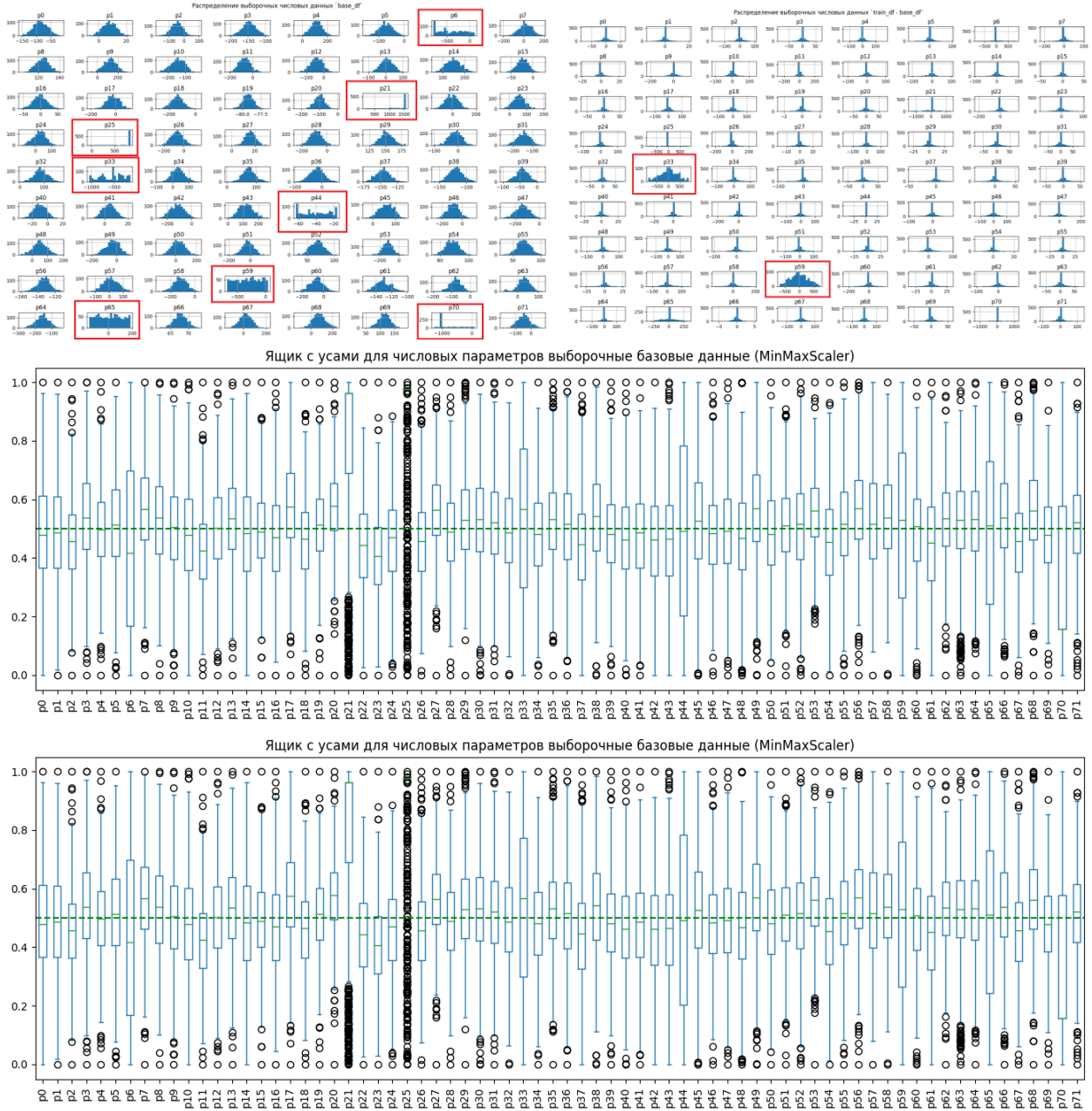
Примечание: предполагается, что метрика расстояния между элементами A и элементами A и B одна и таже.

Этапы решения:

- Загрузка и изучение исходных данных;
- Изучение метрик расстояний для сопоставления A и B;
- Изучение параметров влияющих на расстояние между A и B;
- Оптимизация поиска для большого числа элементов.

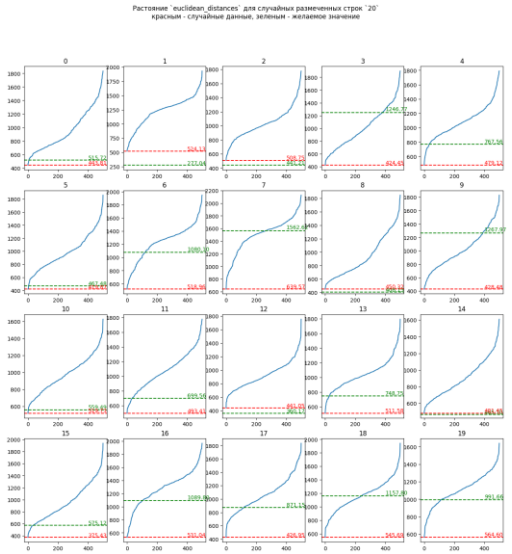
Исходные данные

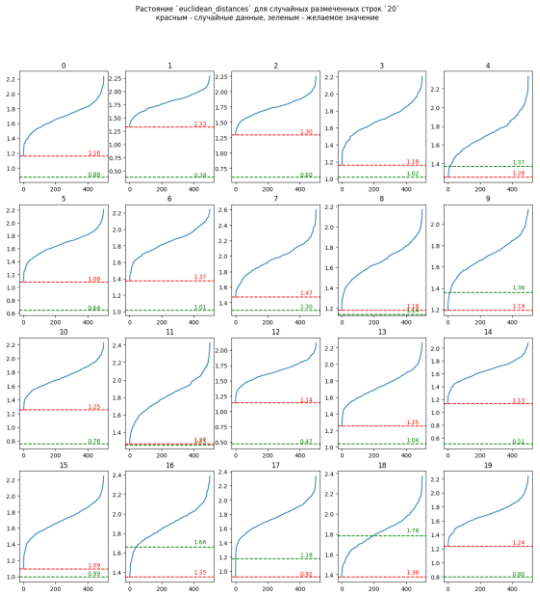
- параметры данных A и B имеют сопоставимый характер;
- параметры имеют нормальное распределение за исключением параметров 6, 21, 25, 33, 44, 59, 65 и 70;
- имеются выбросы и выделяются; параметры 21 имеет смещение, параметр 25 содержит одни выбросы;
- Имеются параметры с числом уникальных значений меньше 30%;
- Разность параметров сосредоточены в 0 и имеют небольшую дисперсию за исключением параметров 33 и 59.

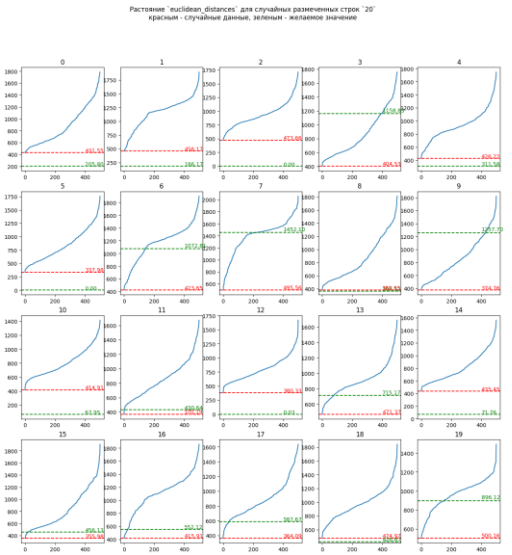


Метрики расстояний, нормирование и поиск параметров

- для 20 значений из b выбиралось 500 случайных значения из a и сравнивались качество метрик расстояний
- при изучении исключались 33 и 59;
- для различных комбинаций параметров 6, 21, 25, 44, 65 и 70 осуществлялась оптимизация метрик расстояний;
- изучены метрики: L1 и L2.







Нормализации данных (`MinMaxScaler`)

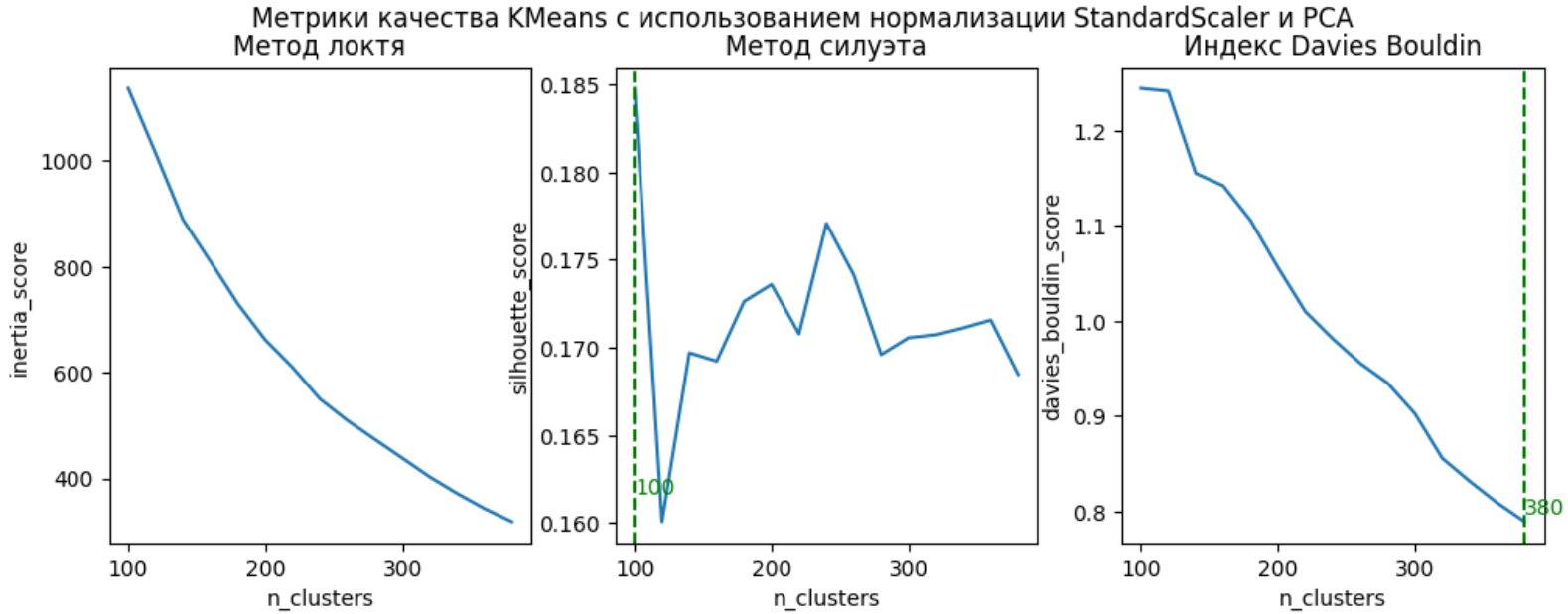
- без удаления параметров Количество ошибочных расстояний: 5 из 20 (25.00%)
- без параметров `p33`, `p59` Количество ошибочных расстояний: 4 из 20 (20.00%)

В сдучае удаеления дополнительных столбов можно получить:

- Количество ошибочных расстояний: 1 из 20 (5.00%)

	columns	correct_values	total	columns_len
1	[p21]	19	20	1
13	[p21, p65]	19	20	2
32	[p21, p25, p65]	19	20	3
2	[p25]	18	20	1
4	[p65]	18	20	1

Исследование кластеризации: KMean



- Исследована метрики качества кластеризации для количества кластеров от 100 до 400 кластеров с шагом 20: метод локтя; метод силуэта; метод Davies-Bouldin.
- Для борьбы с «аномальными» параметрами использовался метод выявления главных компонент (PCA).

Примечание: при увеличение числа кластеров до тысяч метрики начинают улучшаться. Кластеризация зависит от полноты данных.

Примечание: при использовании DBSCAN выявлена сложность подбора параметра φ, в случае использования нормализации данных и т.п.

Оптимизация поиска: Kmean

- Построена функция максимизации качества метрики accuracy@5 в зависимости от гиперпараметров:
  - нормализация данных;
  - Уменьшение размерности (PCA);
  - количество кластеров;
  - Удаляемые параметры.
- Оптимизация осуществлялась на обучающей выборке;
- Наилучший результат на тестовых данных : 0.61

score = 0.5956565223603372

- # exclude_p_columns = ['p21', 'p25', 'p33', 'p65', 'p70']
- # n_clusters = 70
- # n_components = 'mle'
- # scaler = StandardScaler()

score = 0.6155186551865519

- # exclude_p_columns = ['p6', 'p21', 'p25', 'p33', 'p44', 'p59', 'p65', 'p70']
- # n_clusters = 320
- # scaler = StandardScaler()
- # n_components = 'mle'
- # CPU times: user 7min 16s, sys: 2min 17s, total: 9min 34s

score = 0.53498

- # exclude_p_columns = ['p21', 'p25']
- # n_clusters = 320
- # scaler = StandardScaler()
- # n_components = 'mle'
- # CPU times: user 5h 6min 38s, sys: 2h 7min 11s, total: 7h 13min 50s
- # Wall time: 4h 23min 7s

score = 0.3

- # exclude_p_columns = ['p33', 'p59']
- # n_clusters = 320
- # scaler = MinMaxScaler()
- # n_components = 'mle'
- # CPU times: user 5h 6min 38s, sys: 2h 7min 11s, total: 7h 13min 50s
- # Wall time: 4h 23min 7s

Оптимизация поиска: FAISS

- Построена функция максимизации качества метрики accuracy@5 в зависимости от гиперпараметров:
  - нормализация данных;
  - Уменьшение размерности (PCA);
  - количество кластеров;
  - Удаляемые параметры.
- Оптимизация осуществлялась на обучающей выборке;
- Наилучший результат на тестовых данных: 0.78

IndexFlatL2

При удалении параметров `p33` и `p59`:

	drop_columns	scaler	pca_n_components	metric	n_cells	score
0	(p6,)	RobustScaler	mle	1	1024	0.659697
1	(p21,)	RobustScaler	mle	1	1024	0.659697
2	(p25,)	RobustScaler	mle	1	1024	0.659697
3	(p44,)	RobustScaler	mle	1	1024	0.659697
4	(p65,)	RobustScaler	mle	1	1024	0.659697

IndexFlatL1

При удалении параметров p33 и p59:

	drop_columns	scaler	pca_n_components	metric	n_cells	score
0	(p6,)	MinMaxScaler	None	2	1024	0.78772
1	(p21,)	MinMaxScaler	None	2	1024	0.78772
2	(p25,)	MinMaxScaler	None	2	1024	0.78772
3	(p44,)	MinMaxScaler	None	2	1024	0.78772
4	(p65,)	MinMaxScaler	None	2	1024	0.78772

Особенности решения задачи

- При решении задачи на полных данных время расчета значительно оптимизации значительно увеличивается. Например, Sklearn.KMean рассчитывался ~4 часа.
- На ОС windows из коробки не работает FAISS (CPU).
- При использовании бесплатного тарифа сервиса google.colab при полном наборе данных тетрадка завершается из-за нехватки ОЗУ.
- На сервисе google.colab не из коробки запустить FAISS (GPU).

Результат

Лучшая модель использует:

- метрику расстояния L1;
- Параметры для модели все кроме параметров: 6, 33 и 59;
- Нормализация: MinMaxScaler;
- Уменьшение размерности: нет;
- Количество кластеров 1024 (влияет на скорость работы);

Метрика accuracy@5 на проверочных данных:

0.7699677480553975

Предложения по развитие:

- исследовать других метрик расстояний;
- исследование влияния очищенных от выбросов данных;
- настройка FAISS с использованием не точного поиска;
- исследование других алгоритмов поиска ANNOY и т.п.