

Мастерская: Marketplace Matching

Задача:

Есть обезличенные данные по товарам. Нужно на примере 100,000 товаров научиться искать 5 наиболее подходящих к запросу товаров из общей базы с 2,900,000 товаров.

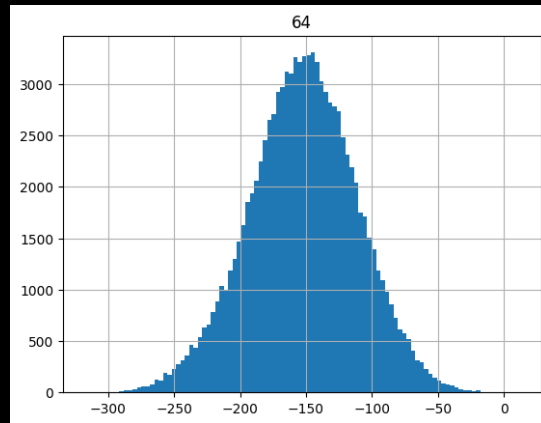
Целевая метрика – $\text{accuracy@5} > 0,8$

Декомпозиция задачи

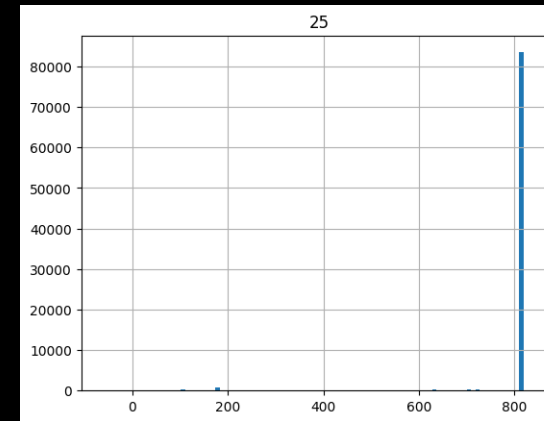
1. Анализ и предобработка данных
2. Грубый поиск (~100 подходящих товаров)
3. Точный поиск – выбрать 5 товаров из кандидатов, отобранных на этапе грубого поиска
4. Принятое допущение: эксперименты на 10% обучающей и валидационной выборки

Признаки

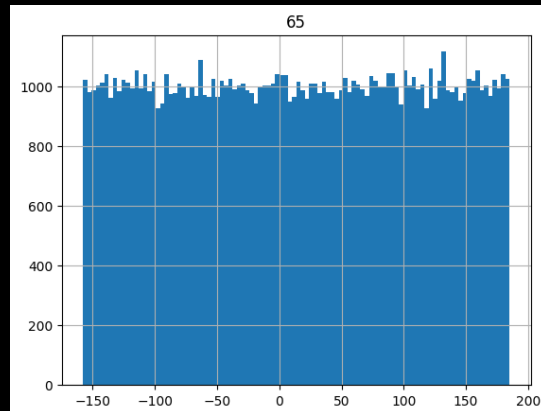
1. Нормально распределенные



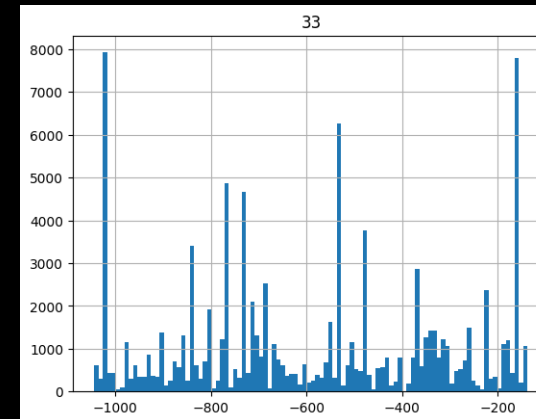
~~3. Константные~~



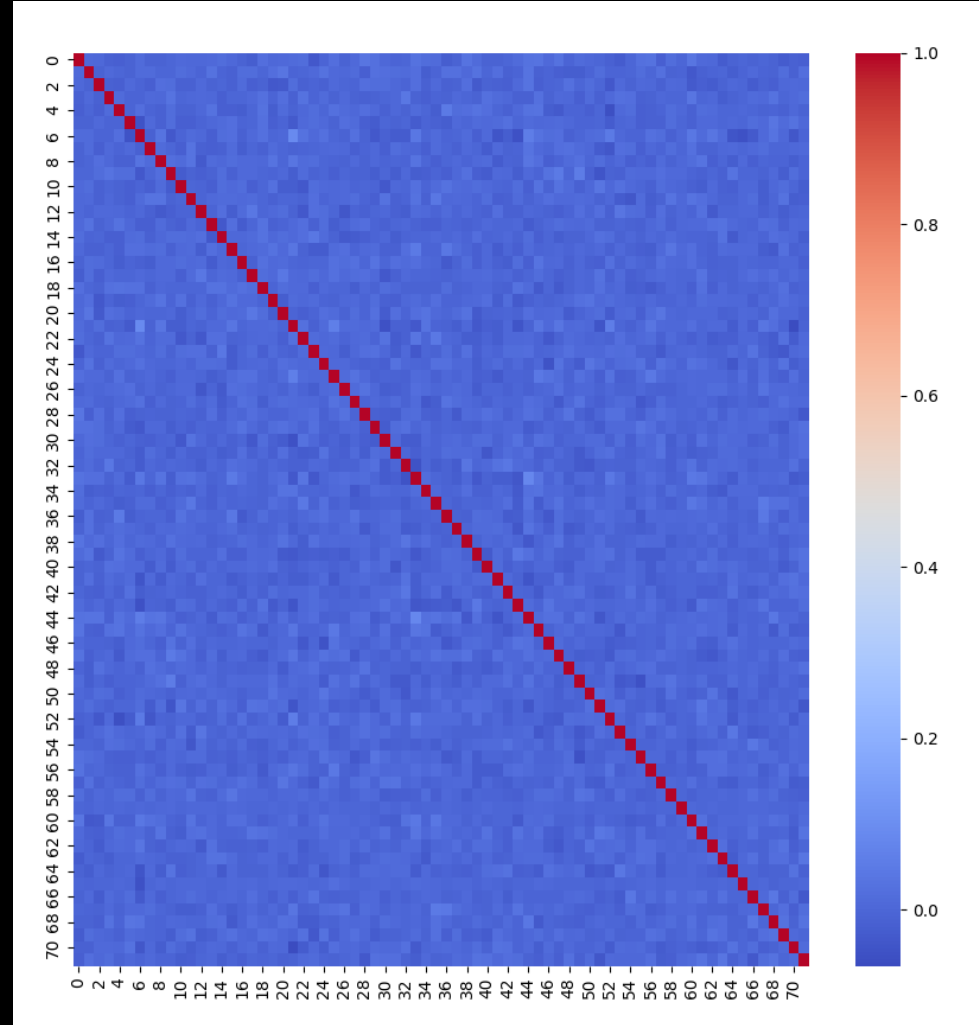
2. Равномерно распределенные



4. Неопикуемые



Корреляция?

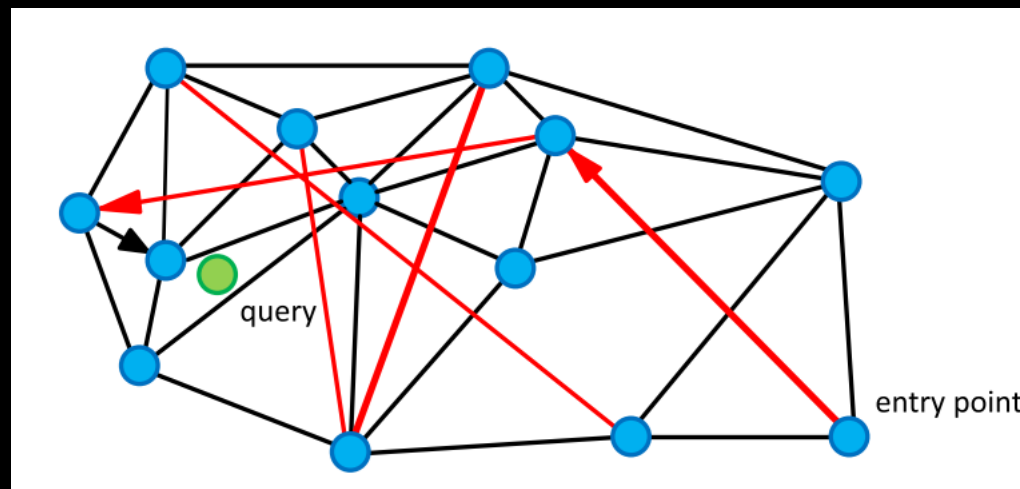


Грубый поиск

- FAISS



- HMSW (nmslib)



Грубый поиск - результаты

nmslib - accuracy@100 < 0.5

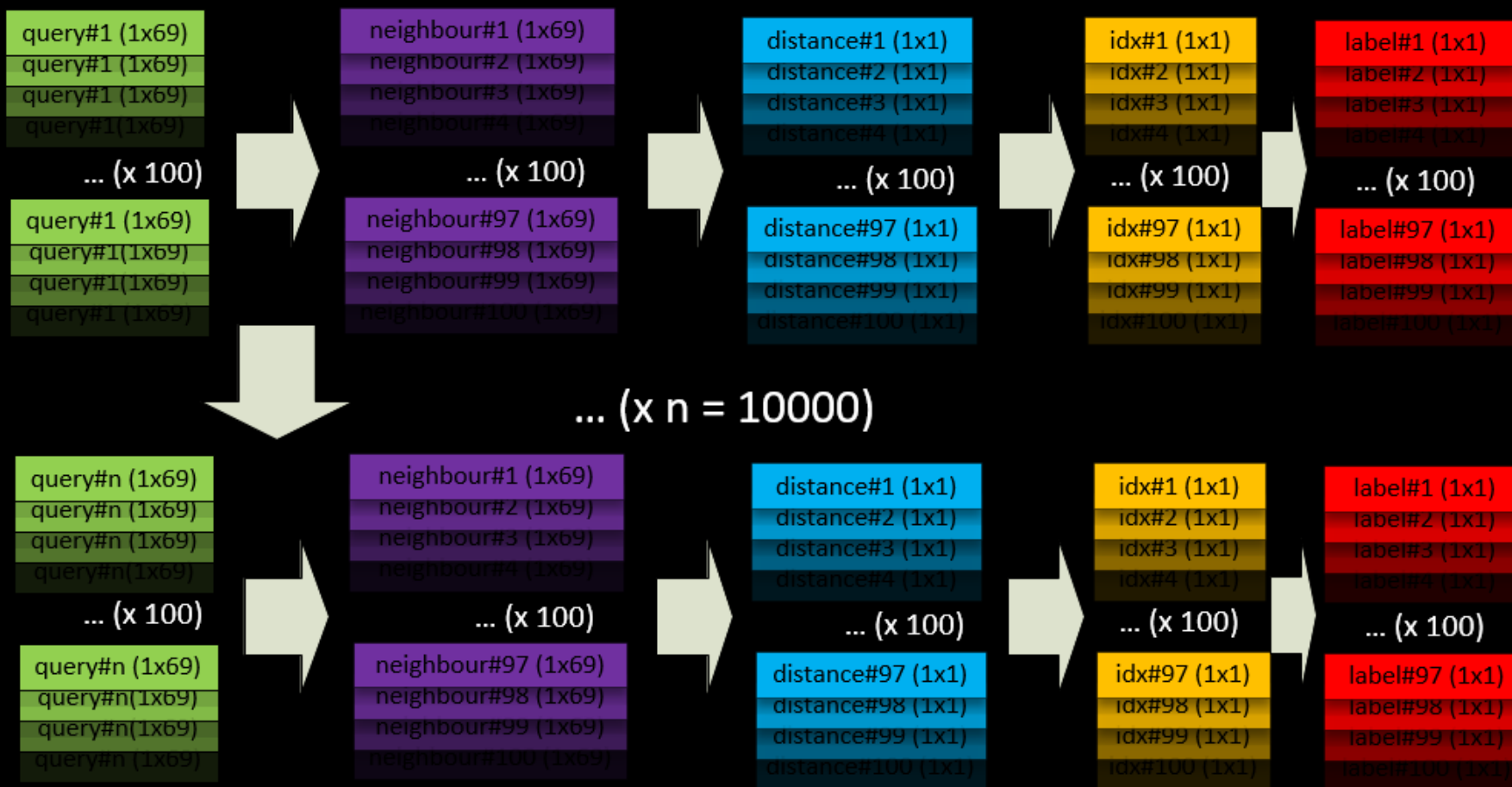
FAISS (CPU) – accuracy@100 < 0,65

FAISS (GPU) – accuracy@100 = 0,8

accuracy@20 = 0,73

n_cells	n_neighbours	Index	result
200	20	IndexFlatL2	0,73
200	20	IndexFlatIp	0,71
200	100	IndexFlatL2	0,79
200	100	IndexFlatIp	0,78
200	200	IndexFlatL2	0,81
200	200	IndexFlatIp	0,80

Сборка матрицы признаков и таргетов




features – 139 x 1,000,000
(для 10% выборки)

Обучение, предсказание, сборка матрицы результатов

Для каждой 100 строк:

- сортировка по убыванию по 'probability'
- сопоставление наличия 'id' в индексах из base, соответствующих значениям из 'true_answer' (для первых 5 записей)



	id	probability	true_answer
0	2192372	0.663457	2676668-base
1	1954150	0.719478	2676668-base
2	814942	0.643366	2676668-base
3	2177660	0.658418	2676668-base
4	2363873	0.660613	2676668-base
...
199995	6486	0.062228	505205-base
199996	220062	0.034175	505205-base
199997	362580	0.035659	505205-base

Точный поиск - результаты

Модель	Accuracy@100	Accuracy@5	Примечание
LogisticRegression	0,74	0,09	Явно видно переобучение на слишком большом количестве признаков
CatBoostRanker	0,74	0,67	Для 2 классов – то же, что и classifier
CatBoostClassifier	0,74	0,68	

А что, так можно было?

Модель	Accuracy@100	Accuracy@5	Примечание
LogisticRegression (n_features=1)	0,74	0,68	В качестве единственного признака использовалось расстояние между векторами. найденное в FAISS
CatBoostRanker	0,74	0,67	Для 2 классов – то же, что и classifier
CatBoostClassifier	0,74	0,68	

Для 20 соседей

Модель	Accuracy@20	Accuracy@5
LogisticRegression	0,73	0,33
LogisticRegression (n_features=1)	0,73	0,68
CatBoostRanker	0,73	0,68
CatBoostClassifier	0,73	0,70

Приложение + Docker

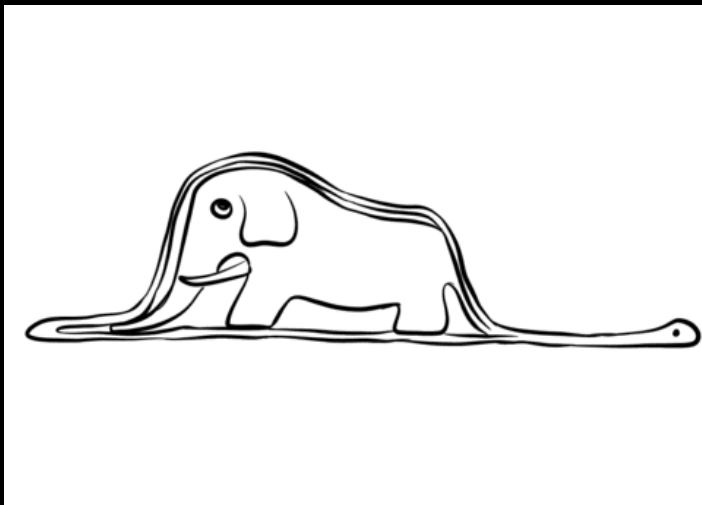
1. Внутри – логистическая регрессия (вся база данных не нужна, только индексы и расстояния из FAISS)
2. Для демонстрации – работает примерно на 3% индексов
3. На входе – строка из файла, проверка на тип данных и количество признаков

```
filename:  "ex_2"  
status:    "success. your query matches with:"  
info:      '{"0":9652,"1":29335,"2":23529,"3":20145,"4":28917}'
```

Проблемы

- Много данных, мало памяти :(
- Непрозрачность параметров FAISS

Batch training, sum_models



```
m_list = []
for step in range(0,10):
    distance, index, candidates = faiss_gen_gpu(features_train, dims, batch_size, step, NeiNum)
    accuracy, labels = acc_at_n(target_train, index, NeiNum, batch_size, step)
    preprocessed_data = collect_data(distance, index, candidates, features_train, labels,
                                    NeiNum, step, batch_size)

    X = preprocessed_data[:, 0:138]
    y = preprocessed_data[:, 140].astype('int')
    w1 = sum(y) / y.shape[0]
    w0 = (y.shape[0] - sum(y)) / y.shape[0]
    del candidates, index, distance, preprocessed_data
    params = {'learning_rate': 0.5,
              'depth': 2,
              'n_estimators': 150,
              'rsm': 1,
              'verbose': False,
              'loss_function': 'Logloss',
              'eval_metric': 'Accuracy',
              'class_weights': (w0, w1),
              'random_state': 2908,
              'task_type': 'GPU'
             }

    model = CatBoostClassifier(**params)
    model.fit(X, y)
    del X, y
    print('STEP---', step)
    m_list.append(model)
```

✓ [14] model = sum_models(m_list)

✓ model.save_model('drive/MyDrive/Colab Notebooks/matching/data/summ.cat')
new_model = CatBoostClassifier()
new_model.load_model('drive/MyDrive/Colab Notebooks/matching/data/summ.cat')

<catboost.core.CatBoostClassifier at 0x79a8622a23e0>

Ответы на вопросы после ревью

Вопрос / замечание	Комментарий
Scaler надо было обучить на features_base, а не на features_train	В данном случае согласен, но всегда ли это так?
Почему тетрадок две?	