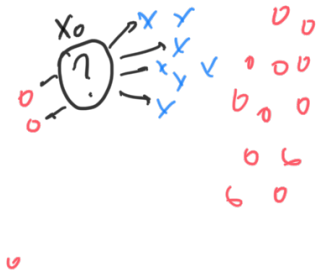
Быстрый поиск соседей

k NN



Поиск ближ. соседей:

① Линейный поиск

```
idx_best, dist_best = 0, +∞
for i in range(l):
    dist = 0
    for j in range(d):
        dist += (x_0j - x_ij)^2
    if dist < dist_best:
        idx_best = i
        dist_best = dist
```

$\simeq l \cdot d$ операций

(долго!!!)

Применения:

1) отбор кандидатов в поиске / рекомендат. системах

в рекоменд.: много айтемов ($|I| > 10^6$)

нужно уменьшить до $\approx 10^3$

p_u - вектор пользователя

кандидаты: 10^3 айтемов

с минимальными $\rho(p_u, q_i)$

в поиске: v_q - вектор запроса

w_i - векторы документов (10^9)

отбираем $i \in I$: $\rho(v_q, w_i) \to \min$

2) image retrieval

ищем картинки, похожие на данную

v_i - векторы для картинок

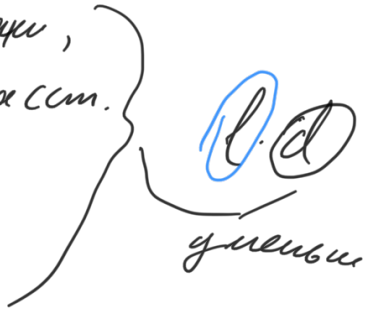
ищем картинки с $\rho(v_0, v_i) \to \min$

Способы ускорения:

① Случайный поиск
(сокращаем X)

② Кластеризовать X
$c_1, \dots c_K$ - центры кластеров
x_0 : ищем ближайший центр
↓
ищем соседей только в этом кластере

③ Найти важные признаки,
сначала считать расст.
только по ним
ИЛИ
сделать PCA



④ kd-trees, ball trees
$d \approx 10-20 \Rightarrow \log l$ операций
при $d \to \infty$ скорость поиска $\to ld$

⑤ Отбор эталонов (STOLP)

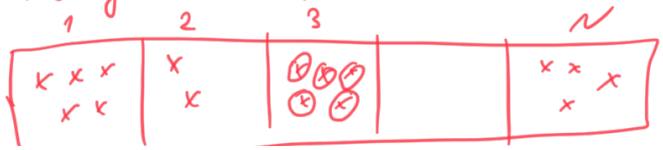
⑥ Приближенный поиск соседей
(ANN)
ищем соседей, которые дальше, чем
ближайшие, не более чем в δ раз

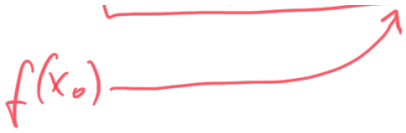
Locality-sensitive hashing (LSH)

$f(x)$ - хэш-функция
$f(x) \in \{1, \dots N\}$ - корзины (buckets)

$f(x_1) = 2$
$f(x_2) = 10$

X - обуч. выборка





Опр. $\mathcal{F}$ – семейство хэш-функций.

$\mathcal{F}$ – (d_1, d_2, p_1, p_2) – чувствительное, если

1) $\rho(x,z) \le d_1 \Rightarrow \mathbb{P}_{f\in\mathcal{F}}[f(x)=f(z)] \ge p_1$

2) $\rho(x,z) \ge d_2 \Rightarrow \mathbb{P}_{f\in\mathcal{F}}[f(x)=f(z)] \le p_2$

Примеры хэш-функций.

1) Косинусное расстояние

$$\frac{\langle x,z\rangle}{\|x\|\,\|z\|}$$

$$f_w(x) = \operatorname{sign}\langle w, x\rangle$$

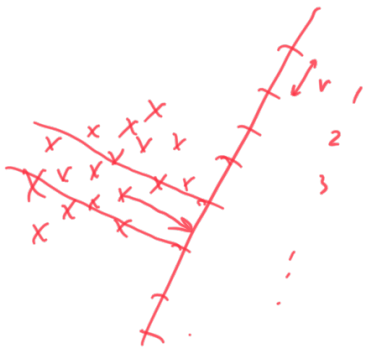
w – случайные векторы

$(d_1, d_2, 1-d_1, 1-d_2)$ – чувствительная

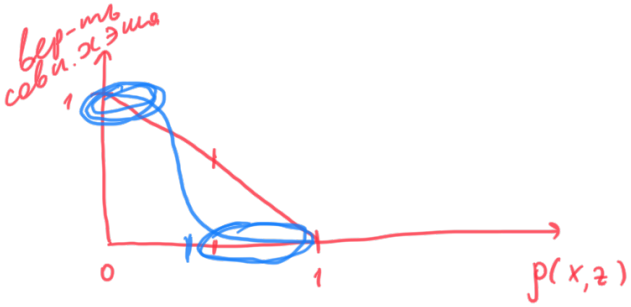
2) Евклидово расстояние

$$\rho(x,z) = \sum_{j=1}^{d} (x_j - z_j)^2$$

$$f_{w,b}(x) = \left\lfloor \frac{\langle w, x\rangle + b}{r} \right\rfloor$$



$w_j \sim \mathcal{N}(0,1)$, $b \sim U[0, r)$



Композиция хэш-функций

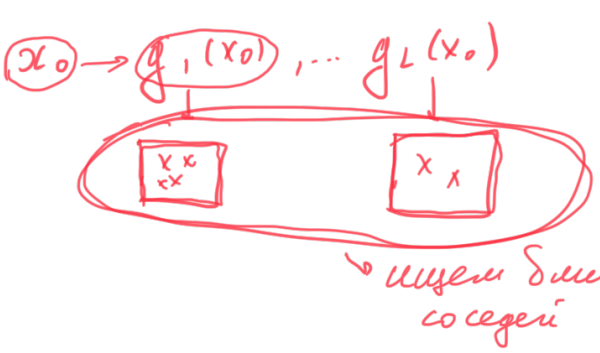
$f_1(x), \dots f_m(x)$ – хэш-функции

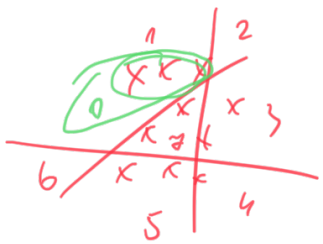
$g(x) = (f_1(x), \dots f_m(x))$ - новый хэш

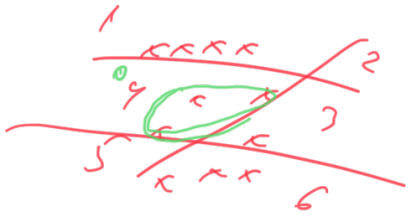
$f_i \in \{0, 1\}$ $\quad m = 3$: 000, 001, 010, 011, ... 111 } 8 штук

$|g| = 2^m$

$g_1(x), \dots g_L(x)$ - случайные композиции







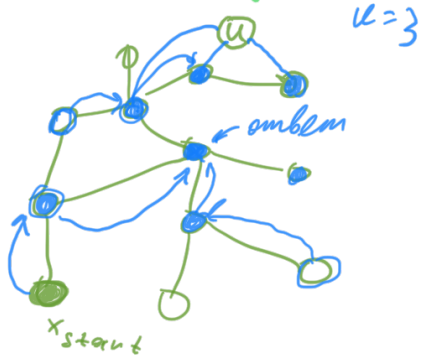
m, L - гиперпараметры

Можно подобрать m и L так, что алгоритм будет находить δ-ближайшего соседа за $\tilde{O}(d l^r \log l)$, $r \approx \frac{1}{\delta}$

$d \sqrt{l} \log l \ll d \cdot l$

NSW (navigable small world)

Строим граф, вершины соответствуют объектам



Поиск соседей

u - запрос

Берем случ. вершину x в графе

в цикле:

если среди соседей x есть x_i : $\rho(u, x_i) < \rho(u, x)$,

то $x = x_1$
(переходим в лучшего соседа)

Будем запускать несколько раз

Построение

Добавление в граф нового объекта и
Запускаем поиск соседей m раз
$S = \{x_1, \dots x_m\}$ - кандидаты
дополняем S соседями $x_1, \dots x_m$
соединяем и ребрами с k
ближ. соседями из S

HNSW $O(\log l)$

'hierarchical