Хеширование признаков

Илья Трофимов

22.09.2016

Машинное обучение и большие данные
ФИВТ, осень 2016

Признаки

Машинное обучение с учителем:
Дана выборка $(\mathbf{x}_i, y_i)_{i=1}^n, \quad \mathbf{x}_i \in X = \mathbb{R}^p, y_i \in Y$

построить функцию $f : X \to Y$

Признаки

На самом деле:

Машинное обучение с учителем:

по выборке $(o_i, y_i)_{i=1}^n$, $o_i \in O$ (множество объектов), $y_i \in Y$
сгенерировать p признаков с помощью функций $f_1, \ldots, f_p$

$$f_j : O \to \mathbb{R}$$
$$\mathbf{x}_i = [f_1(o_i), \ldots, f_p(o_i)]$$

построить функцию $f : X \to Y$

Пространства признаков высокой размерности

Пример 1. Классификация текстов. Представление в виде мешка слов.
Признаки - N-граммы.
Признаки - N-граммы с пропусками. (K-skip-N-grams)

Пространства признаков высокой размерности

Пример 1. Классификация текстов. Представление в виде мешка слов.
Признаки - N-граммы.
Признаки - N-граммы с пропусками. (K-skip-N-grams)

Текст: ABCDE
2-граммы: AB, BC, CD, DE
1-скип-2-граммы: A?C, B?D, C?E

Пространства признаков высокой размерности

Пример 1. Классификация текстов. Представление в виде мешка слов.
Признаки - N-граммы.
Признаки - N-граммы с пропусками. (K-skip-N-grams)

Текст: ABCDE
2-граммы: AB, BC, CD, DE
1-скип-2-граммы: A?C, B?D, C?E

Размерность быстро растет с ростом N

Пространства признаков высокой размерности

Пример 1. Классификация текстов. Представление в виде мешка слов.
Признаки - N-граммы.
Признаки - N-граммы с пропусками. (K-skip-N-grams)

Текст: ABCDE
2-граммы: AB, BC, CD, DE
1-скип-2-граммы: A?C, B?D, C?E

Размерность быстро растет с ростом N

Пример 2. Предсказание вероятности клика. Комбинации категориальных признаков.
UserID $\sim 10^6$, AdID $\sim 10^6$

Пространства признаков высокой размерности

Пример 1. Классификация текстов. Представление в виде мешка слов.
Признаки - N-граммы.
Признаки - N-граммы с пропусками. (K-skip-N-grams)

Текст: ABCDE
2-граммы: AB, BC, CD, DE
1-скип-2-граммы: A?C, B?D, C?E

Размерность быстро растет с ростом N

Пример 2. Предсказание вероятности клика. Комбинации категориальных признаков.
$\text{UserID} \sim 10^6, \text{AdID} \sim 10^6$
$\text{UserID} \times \text{AdID} \sim 10^{12}$

Генерация признаков

$$\mathbf{x}_i = [f_1(o_i), \ldots, f_p(o_i)]$$

Методы машинного обучения принимают на вход вектор признаков $\mathbf{x} \in \mathbb{R}^p$

Генерация признаков

$$\mathbf{x}_i = [f_1(o_i), \ldots, f_p(o_i)]$$

Методы машинного обучения принимают на вход вектор признаков $\mathbf{x} \in \mathbb{R}^p$

Подход 1. Пронумеровать признаки, проставить $f_j(o_i)$ в обучении и тесте.

Генерация признаков

$$\mathbf{x}_i = [f_1(o_i), \ldots, f_p(o_i)]$$

Методы машинного обучения принимают на вход вектор признаков $\mathbf{x} \in \mathbb{R}^p$

Подход 1. Пронумеровать признаки, проставить $f_j(o_i)$ в обучении и тесте.

1. Нумерация может идти дольше, чем онлайн обучение
2. Увеличение размера датасета
3. Получается пространство большой размерности
4. Не онлайн
5. Необходимо использовать словарь `<признак, индекс>` при прогнозе

Генерация признаков

Хранить параметры модели в ассоциативном массиве (хеш-таблица, дерево): `map<признак, вес>`

$$\mathbf{x}_i = [f_1(o_i), \ldots, f_p(o_i)]$$

Методы машинного обучения принимают на вход вектор признаков $\mathbf{x} \in \mathbb{R}^p$

Генерация признаков

Хранить параметры модели в ассоциативном массиве (хеш-таблица, дерево): `map<признак, вес>`

$$\mathbf{x}_i = [f_1(o_i), \ldots, f_p(o_i)]$$

Методы машинного обучения принимают на вход вектор признаков $\mathbf{x} \in \mathbb{R}^p$

Объект: o = ABCDE
Метка: y
Признаки: $\mathbf{x}$ = AB, BC, CD, DE
Шаг стохастического градиента:

1. for *key* in { AB, BC, CD, DE }:
2. $w[key] = w[key] - \eta \frac{\partial L(f_w(\mathbf{x}), y)}{\partial w[key]}$

Генерация признаков

Подход 2. Генерировать признаки «на лету», использовать ассоциативный массив.

1. Нумерация не используется
2. Размера датасета не увеличивается
3. Получается пространство большой размерности
4. Онлайн
5. Необходимо использовать словарь `<признак, вес>` при прогнозе

Хеш-таблица

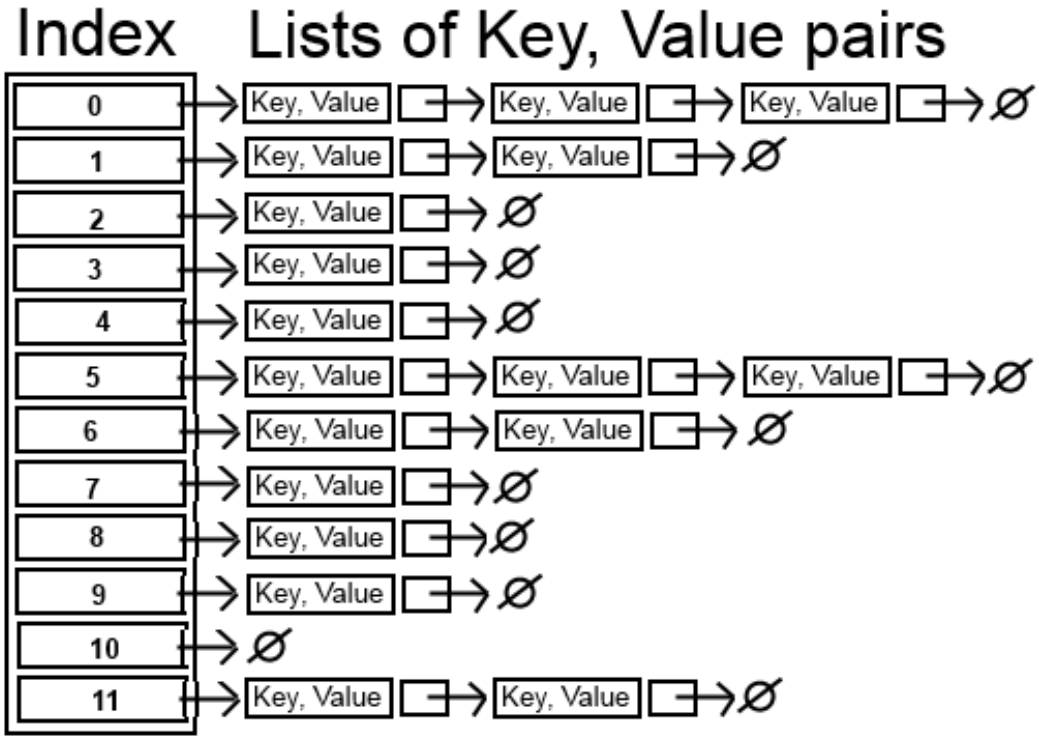
Хеш-функция: признак (строка) $\rightarrow \{1, \ldots 2^b\}$

Отображение должно быть достаточно «равномерным».

Хеш-таблица

Хеш-функция: признак (строка) $\to \{1, \ldots 2^b\}$
Отображение должно быть достаточно «равномерным».

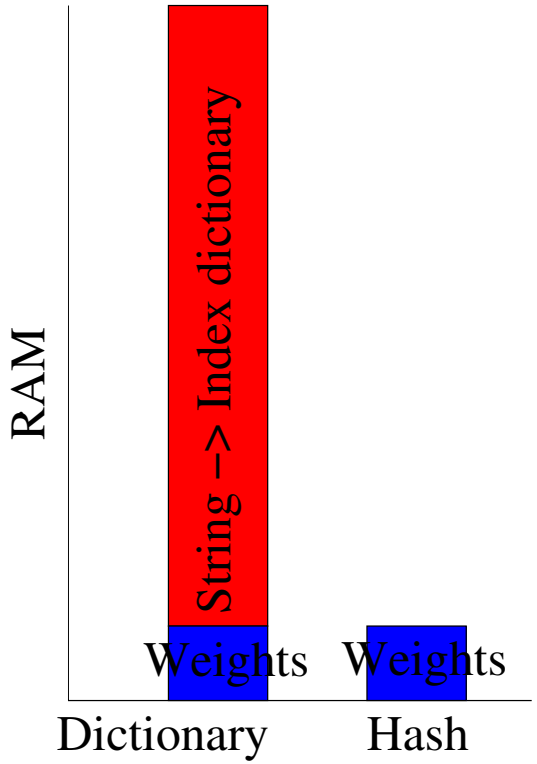
Хеш-таблица, v1 = Хеш-функция + таблица размера 2^b, в ячейках таблицы храним списки (ключ, значение), если коллизии разрешаются методом цепочек



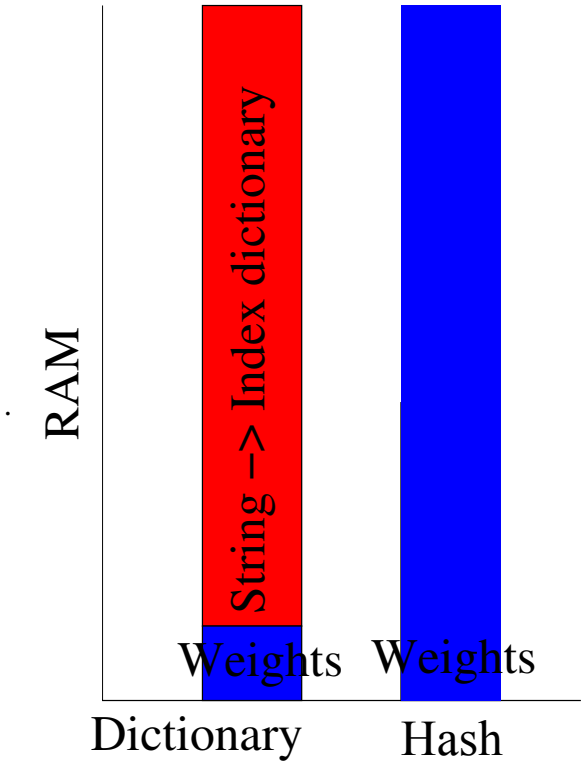
Как по признаку найти значение веса

Хеш-таблица, v2 - с коллизиями.

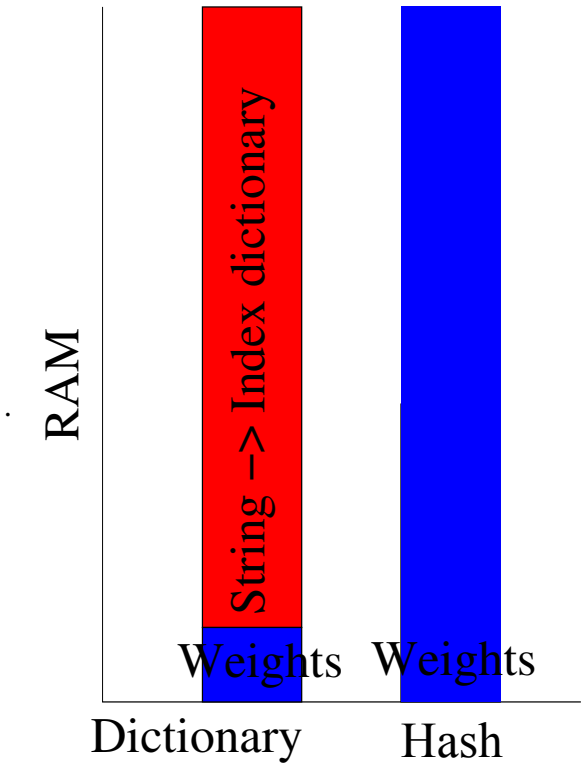
Сравнение методов хранения весов



Сравнение методов хранения весов



Сравнение методов хранения весов



Чем больше храним весов - тем лучше!

Коллизии

Основным возражением против хеширования являются коллизии.

Коллизии

Основным возражением против хеширования являются коллизии.

1. Актуально только на датасетах с небольшим числом признаков. Если признаков много, то среди них, как правило, есть сильно коррелированные (избыточность). Часть из них избежит коллизий.
2. На практике можно подобрать число бит в хеш-таблице b, так что при его увеличении качество предсказания не улучшается. Значит при данном значении b коллизии не вредят.

Хеш-ядро

Пусть o - исходный объект обучающей выборки.

Хеш-ядро

Пусть o - исходный объект обучающей выборки.
Обозначим $\phi(o) \in \mathbb{R}^{2^b}$ - вектор признаков после хеширования.

Хеш-ядро

Пусть o - исходный объект обучающей выборки.
Обозначим $\phi(o) \in \mathbb{R}^{2^b}$ - вектор признаков после хеширования.

$$K(o, w) = (\phi(o), w) \; - \; \text{хеш-ядро}$$

Решающее правило

$$K(o, w) > a$$

в новом признаковом пространстве будет линейным:

$$(\phi(o), w) > a$$

Хеш-ядро

Пусть o - исходный объект обучающей выборки.
Обозначим $\phi(o) \in \mathbb{R}^{2^b}$ - вектор признаков после хеширования.

$$K(o, w) = (\phi(o), w) \; - \; \text{хеш-ядро}$$

Решающее правило

$$K(o, w) > a$$

в новом признаковом пространстве будет линейным:

$$(\phi(o), w) > a$$

Представление объекта o в признаковом пространстве не нужно хранить явно, можно вычислять «на лету».

Hashing trick

Дано: $\mathbf{x} \in \mathbb{R}^n$ (n - большое),
хеш-функции $h : \mathbb{N} \to \{1, \ldots, 2^b\}$, $\xi : \mathbb{N} \to \{-1, +1\}$

$$\phi_i^{h,\xi}(\mathbf{x}) = \sum_{j:h(j)=i} \xi(j) x_j$$

$$(\mathbf{x}, \mathbf{x}')_\phi \stackrel{\text{def}}{=} \sum_{i=1}^{2^b} \phi_i^{h,\xi}(\mathbf{x}) \phi_i^{h,\xi}(\mathbf{x}')$$

Утверждение 1. Хеш-ядро не смещено:

$$\mathbb{E}[(\mathbf{x}, \mathbf{x}')_\phi] = (\mathbf{x}, \mathbf{x}')$$

Утверждение 2. Оценка на дисперсию если $\|\mathbf{x}\|_2 = \|\mathbf{x}'\|_2 = 1$:

$$\mathbb{D}[(\mathbf{x}, \mathbf{x}')_\phi] = O\left(\frac{1}{2^b}\right)$$

Можно использовать как способ понижения размерности для разреженных $\mathbf{x}$.

Генерация признаков

Подход 3. Генерировать признаки «на лету», использовать хеш-таблицу с коллизиями.

1. Нумерация не используется
2. Размера датасета не увеличивается
3. Размер пространства признаков фиксирован
4. Онлайн
5. Для прогноза нужен только вектор размера 2^b

Пример: персонализованная фильтрация спама

1. $3.2 * 10^6$ размеченных писем.
2. 433167 пользователей.
3. $\sim 40 * 10^6$ уникальных слов.

Как обучить персонализованный спам-фильтр, который был бы лучше неперсонализованного?

Пример: персонализованная фильтрация спама

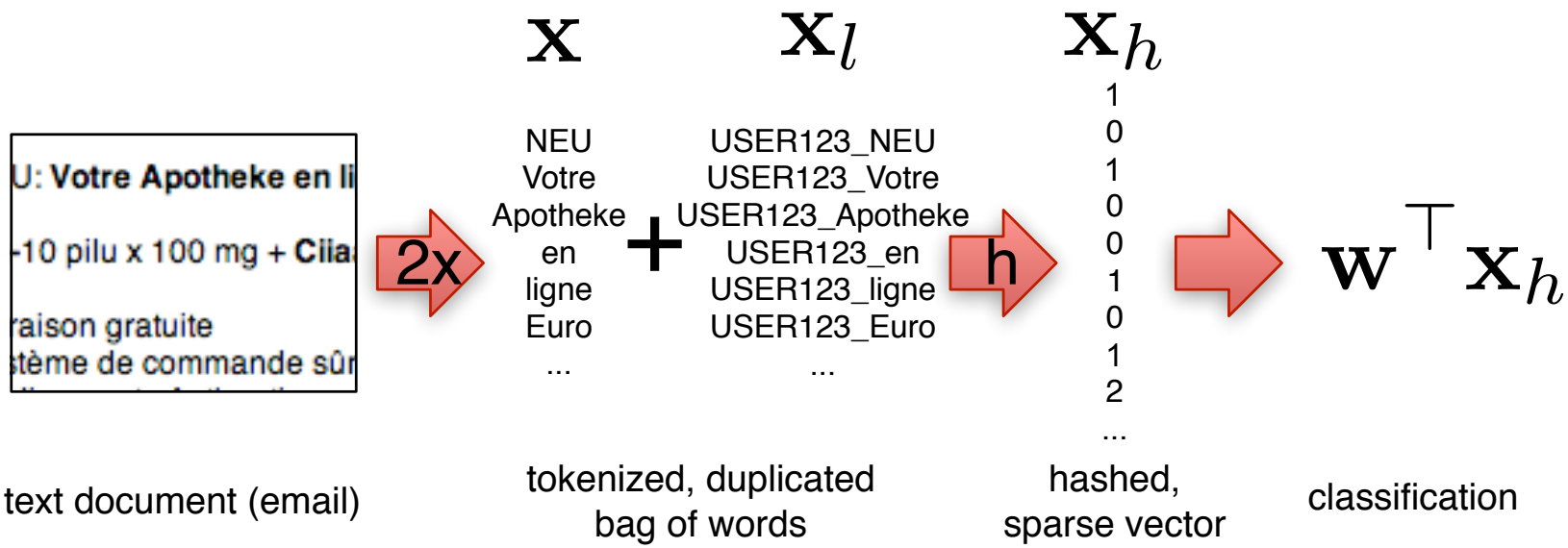
1. $3.2 * 10^6$ размеченных писем.
2. 433167 пользователей.
3. $\sim 40 * 10^6$ уникальных слов.

Как обучить персонализованный спам-фильтр, который был бы лучше неперсонализованного?

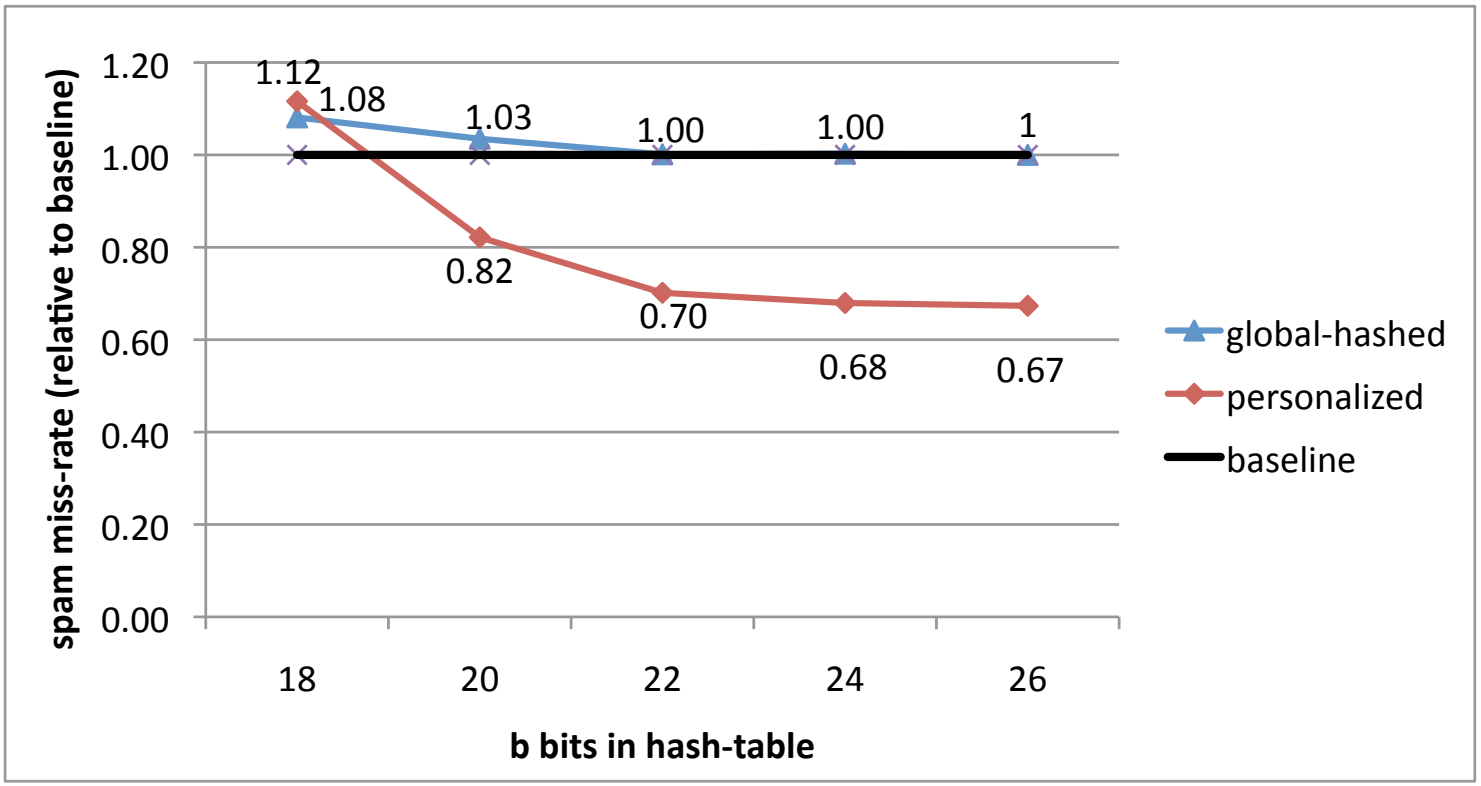
Плохое решение: Обучить общий фильтр (для всех пользователей) + 433167 персональных фильтров, использующих hashmap для хранения весов признаков. Это потребует $433167 * 40 * 10^6 * 4 \sim 70$ Терабайт.

Используем хеширование

Используем хеширование для прогноза: $\langle w, \phi(x) \rangle + \langle w, \phi_u(x) \rangle$



Результаты



2^{26} параметров $=$ 64M весов $=$ 256MB RAM.

В x270K более эффективное использование RAM.

Список литературы

hash Shi, Q., Petterson, J., Dror, G., Langford, J., Smola, A., Vishwanathan, S. V. N. (2009). Hash kernels for structured data. The Journal of Machine Learning Research, 10, 2615-2637.

spam Kilian Weinberger, Anirban Dasgupta, Josh Attenberg, John Langford, A. S. (2009). Feature Hashing for Large Scale Multitask Learning. In ICML.

dna Sonnenburg, S. (2010). COFFIN : A Computational Framework for Linear SVMs, in ICML.

ad Chapelle O., Manavoglu, E., Rosales R. Simple and scalable response prediction for display advertising