Stichproben aus Datenströmen

1. Stichproben fester Größe

Reservoir Sampling

(nach J.S. Vitter, 1985)

Eingabe:

Ein Datenstrom $x_1, x_2, x_3, \dots$ unbekannter Länge

Ziel:

Ein randomisierter Algorithmus, der sich eine Zahl $i \geq 1$ und einen Wert y merkt,

so dass für jeden Zeitpunkt $n \geq 1$ gilt:

Sind i und y die beiden vom Algorithmus gespeicherten Werte nach dem Lesen der ersten n Werte $x_1, x_2, \dots, x_n$ des Datenstroms, so ist

- $i \in \{1, \dots, n\}$
- $y = x_i$
- für jedes feste $j \in \{1, \dots, n\}$: $\quad Pr(i=j) = \frac{1}{n}$

D.h.: Der Algorithmus soll zufällig eins der Elemente $x_1, \dots, x_n$ auswählen — jedes davon mit derselben Wahrscheinlichkeit $\frac{1}{n}$.

Beobachtung:

Falls n von Anfang an bekannt ist, ist das leicht zu erreichen:

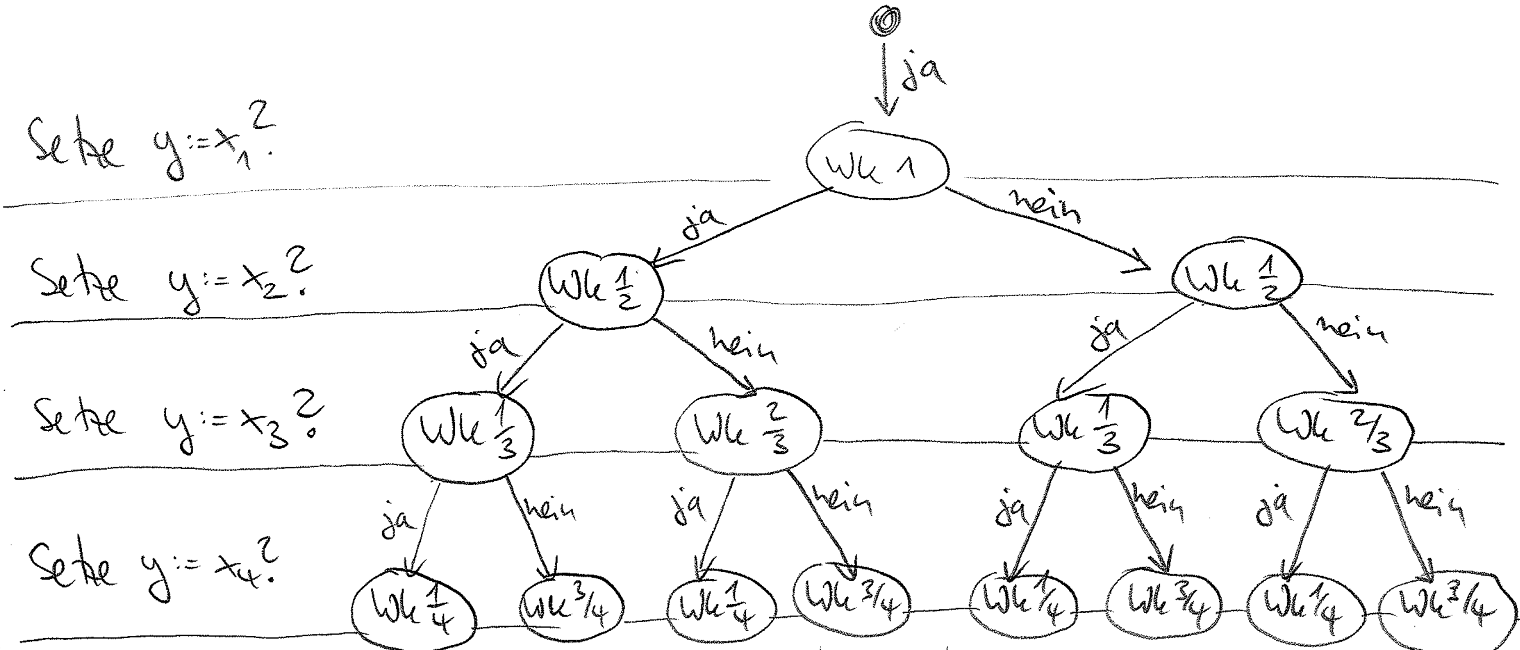
Vor Beginn der Verarbeitung des Datenstroms wählt der Algorithmus einfach eine Zahl $i \in \{1, \dots, n\}$ zufällig (gleichverteilt) aus und speichert dann in y den Wert x_i.

Problem:

Wir kennen n nicht!

Unser Algorithmus soll eine Datenstruktur bereitstellen, die beim Lesen jedes neuen Datenstrom-Elements aktualisiert wird — und zwar so, dass nach dem Lesen der ersten n Datenstrom-Elemente die Datenstruktur einen Index $i \in \{1, \dots, n\}$ und den zugehörigen Wert $y = x_i$ enthält, wobei i eine zufällig aus $\{1, \dots, n\}$ gewählte Zahl ist.

Illustration zur Vorgehensweise des Algorithmus Reservoir-Sampling-1 beim Verarbeiten von $x_1, x_2, x_3, x_4, \ldots$:



Inhalt von y:	x_4	x_3	x_4	x_2	x_4	x_3	x_4	x_1
Wk dafür:	$\frac{1}{2}\cdot\frac{1}{3}\cdot\frac{1}{4} = \frac{1}{24}$	$\frac{1}{2}\cdot\frac{1}{3}\cdot\frac{3}{4} = \frac{3}{24}$	$\frac{1}{2}\cdot\frac{2}{3}\cdot\frac{1}{4} = \frac{2}{24}$	$\frac{1}{2}\cdot\frac{2}{3}\cdot\frac{3}{4} = \frac{6}{24}$	$\frac{1}{2}\cdot\frac{1}{3}\cdot\frac{1}{4} = \frac{1}{24}$	$\frac{1}{2}\cdot\frac{1}{3}\cdot\frac{3}{4} = \frac{3}{24}$	$\frac{1}{2}\cdot\frac{2}{3}\cdot\frac{1}{4} = \frac{2}{24}$	$\frac{1}{2}\cdot\frac{2}{3}\cdot\frac{3}{4} = \frac{6}{24}$

Insgesamt:

Wk für x_1: $\frac{6}{24} = \frac{1}{4}$

x_2: $\frac{6}{24} = \frac{1}{4}$

x_3: $\frac{3}{24} + \frac{3}{24} = \frac{6}{24} = \frac{1}{4}$

x_4: $\frac{1}{24} + \frac{2}{24} + \frac{1}{24} + \frac{2}{24} = \frac{6}{24} = \frac{1}{4}$

Datenstrom-Algorithmus "Reservoir-Sampling-1":

Initialisierung:

Beim Lesen des ersten Datenstrom-Elements x_1 setze
- $i := 1$
- $y := x_1$
- $n := 1$ % Länge des bisher verarbeiteten Teils des Datenstroms

Aktualisierung:

Beim Lesen des nächsten Datenstrom-Elements x_{n+1} tue Folgendes:
- Wähle zufällig, gleichverteilt eine Zahl $r \in \{1, 2, \ldots, n+1\}$
- Falls $r = n+1$, so setze $i := n+1$ und $y := x_{n+1}$
- Setze $n := n+1$

Satz (zum Algorithmus Reservoir-Sampling-1)

Zu jedem Zeitpunkt $n \geq 1$ gilt für die Werte i und y, die der Algorithmus Reservoir-Sampling-1 nach dem Verarbeiten der ersten n Datenstrom-Elemente $x_1, \ldots, x_n$ gespeichert hat: $\Pr(i=j) = \frac{1}{n}$, für jedes feste $j \in \{1, \ldots, n\}$.

Außerdem nutzt der Algorithmus während der Verarbeitung der ersten n Datenstrom-Elemente nur $\ell + O(\log n)$ viele Speicherbits, wobei ℓ eine obere Schranke für die zur Speicherung eines einzelnen Elements aus $\{x_1, \dots, x_n\}$ nötigen Bits ist.

Beweis:

Die Aussage über den verwendeten Speicherplatz gilt offensichtlicherweise.

Zum Nachweis, dass $\Pr(i=j) = \frac{1}{n}$ für jedes feste $j \in \{1, \dots, n\}$ gilt, beachte Folgendes:

Für ein festes $j \in \{1, \dots, n\}$ enthält am Ende der Verarbeitung von $x_1, \dots, x_n$ die Variable i den Wert j genau dann, wenn gilt:

1.) bei der Verarbeitung von x_j wurde $r = j$ gewählt, und

2.) bei der Verarbeitung von $x_{j+1}, \dots, x_n$ wurde jedesmal $r \neq j+1, \dots, r \neq n$ gewählt.

Die Wahl entsprechend 1) geschieht mit Wahrscheinlichkeit $\frac{1}{j}$ (da r zufällig, gleichverteilt aus $\{1, \dots, j\}$ gewählt wird!

Die Wahl entsprechend 2) geschieht für (17)

x_{j+1} mit Wk $\frac{j}{j+1}$ (da r aus $\{1, \ldots, j+1\}$ gewählt)

x_{j+2} .... $\frac{j+1}{j+2}$ (..... $\{1, \ldots, j+2\}$ ...)

$\vdots$

x_n .... $\frac{n-1}{n}$ (..... $\{1, \ldots, n\}$ ...)

Da die Zufallszahlen jeweils unabhängig voneinander gewählt werden, geschieht 1) <u>und</u> 2) mit folgender Wahrscheinlichkeit:

$$\frac{1}{j} \cdot \frac{j}{j+1} \cdot \frac{j+1}{j+2} \cdot \ldots \cdot \frac{n-1}{n} = \frac{1}{n}$$

$\square$

Der Algorithmus Reservoir-Sampling-1 liefert uns eine Datenstruktur, die aus einem Datenstrom eine Stichprobe der Größe 1 zufällig, gleichverteilt auswählt.

Dies kann man verallgemeinern für Stichproben einer festen Größe $s \geq 1$:

Sei $s \geq 1$ eine fest gewählte Stichprobengröße.

Ziel:

Ein randomisierter Algorithmus, der sich eine Menge S von maximal s Tupeln der Form $t = (i, y)$ merkt, so dass für jeden Zeitpunkt $n \geq 1$ gilt:

Ist S die vom Algorithmus gespeicherte Menge nach dem Lesen der ersten n Werte $x_1, \dots, x_n$ des Datenstroms, so gilt:

- falls $n \leq s$, so ist
$$S = \{ (1, x_1), \dots, (n, x_n) \}$$
- falls $n > s$, so ist
$|S| = s$ und $S \subseteq \{ (1, x_1), \dots, (n, x_n) \}$,
und für jede feste s-elementige Menge $M \subseteq \{ (1, x_1), \dots, (n, x_n) \}$ gilt:
$$\Pr(S = M) = \frac{1}{\binom{n}{s}}$$

D.h.: Der Algorithmus soll zufällig eine s-elementige Teilmenge von $(1, x_1), \dots, (n, x_n)$ auswählen — jede davon mit derselben Wahrscheinlichkeit. D.h.: Jedes Tupel (j, x_j), für $j \in \{1, \dots, n\}$, gehört mit Wk $\frac{s}{n}$ zur Stichprobe S.

Datenstrom-Algorithmus "Reservoir-Sampling-s", für $s \in \mathbb{N}_{\geq 1}$

Initialisierung:

$S := \emptyset$

Für $n := 1$ bis s tue Folgendes:

beim Lesen des n-ten Datenstrom-Elements x_n füge das Tupel (n, x_n) in die Menge S ein.

Aktualisierung für $n \geq s$

Beim Lesen des nächsten Datenstrom-Elements x_{n+1} tue Folgendes:

- Wähle zufällig, gleichverteilt eine Zahl $r \in \{1, 2, \ldots, n+1\}$
- Falls $r \leq s$, so
  wähle zufällig, gleichverteilt ein Tupel $t \in S$ aus,
  entferne dieses Tupel aus S und
  füge als Ersatz das Tupel $(n+1, x_{n+1})$ in S ein
- Setze $n := n+1$

Satz (zum Algorithmus Reservoir-Sampling-s)

Zu jedem Zeitpunkt $n \geq s$ gilt für die Menge S, die der Algorithmus Reservoir-Sampling-s nach dem Verarbeiten der ersten n Datenstrom-Elemente $x_1, \dots, x_n$ gespeichert hat:

$$\Pr(S = M) = \frac{1}{\binom{n}{s}}, \text{ für jede feste}$$

s-elementige Menge $M \subseteq \{(1, x_1), \dots, (n, x_n)\}$.

Außerdem nutzt der Algorithmus während der Verarbeitung der ersten n Datenstrom-Elemente nur $s \cdot (\ell + \log n) + O(\log n)$ viele Speicherbits, wobei ℓ eine obere Schranke für die zur Speicherung eines einzelnen Elements aus $\{x_1, \dots, x_n\}$ nötigen Bits ist.

Beweis: Übung!

2. Stichproben variabler Größe

Ziel: Für eine feste Zahl $p > 0$ soll die Größe der Stichprobe etwa

$$\frac{1}{p} \cdot (\text{Länge des bisher gesehenen Teils des Datenstroms})$$

sein.

Ein Beispiel:

Eine Suchmaschine verarbeitet Datenströme, die aus Tupeln der Form

(NutzerID, Anfrage, Zeitpunkt)

bestehen.

Es soll folgende Frage beantwortet werden:

(*): Welcher Anteil von Anfragen wurde im letzten Monat mind 2mal vom gleichen Nutzer gestellt?

Um diese Frage zu beantworten soll nur ungefähr $\frac{1}{10}$ des Datenstroms gespeichert werden.

Naive Lösung:

Beim Verarbeiten jedes Tupels t des Datenstroms: erzeuge eine Zufallszahl $z \in \{1, \ldots, 10\}$ und speichere t nur dann in der Stichprobe S ab, wenn $z = 1$ ist.

Nach dem Verarbeiten von n Tupeln hat die Stichprobe dann im Erwartungswert die Größe $\frac{n}{10}$.

Problem:

Wenn wir die Frage ⊛ auf den in S gespeicherten Tupeln beantworten, erhalten wir einen grundlegend anderen Wert als wenn wir die Frage ⊛ auf dem gesamten Datenstrom $t_1, t_2, \ldots, t_n$ beantworten.

Denn:

Betrachte das Szenario, bei dem $t_1, t_2, \ldots, t_n$ von der folgenden Form ist:

- es gibt k verschiedene Nutzer
- jeder der k Nutzer hat
  - a Anfragen genau 1mal, und
  - b Anfragen genau 2mal gestellt

Es gilt also:

- $n = k \cdot (a + 2b)$
- Antwort auf Frage ⊛ auf dem gesamten Datenstrom:
  - für jeden einzelnen Nutzer: $\frac{b}{a+b}$

    (denn: Der Nutzer stellte $a+b$ verschiedene Anfragen; b davon doppelt)
  - insgesamt: $\frac{k \cdot b}{k \cdot (a+b)} = \frac{b}{a+b}$

    (denn: Jeder der k Nutzer stellte $a+b$ verschiedene Anfragen (also insges. $k(a+b)$ Anfragen); b davon doppelt (also insges $k \cdot b$ doppelte Anfragen))

- Antwort auf Frage ⊛ aus der Stichprobe S':
  - S' enthält nur ca $\frac{1}{10}$ der Tupel des Datenstroms – jedes davon mit Wk $\frac{1}{10}$.

    Für jeden einzelnen Nutzer enthält S' daher
    - ungefähr $\frac{a}{10}$ der 1-mal vom Nutzer gestellten Anfragen
    - jede der b 2-mal vom Nutzer gestellten Anfragen
      - mit Wk $\frac{1}{10} \cdot \frac{1}{10} = \frac{1}{100}$ 2-mal
      - mit Wk $\frac{1}{10} \cdot \frac{9}{10} + \frac{9}{10} \cdot \frac{1}{10} = \frac{18}{100}$ 1-mal
      - mit Wk $\frac{9}{10} \cdot \frac{9}{10} = \frac{81}{100}$ 0-mal
    - Im Erwartungswert enthält S' also

      $\frac{a}{10} + \frac{18 \cdot b}{100}$ Anfragen des Nutzers genau 1mal

      und $\frac{1 \cdot b}{100}$ Anfragen des Nutzers genau 2mal
  - Somit ist für jeden einzelnen Nutzer die Antwort auf Frage ⊛ im Erwartungswert

$$\frac{\frac{b}{100}}{\frac{a}{10} + \frac{18b}{100} + \frac{b}{100}} = \frac{\frac{b}{100}}{\frac{10a + 19b}{100}} = \frac{b}{10a + 19b},$$

    wenn sie auf der Stichprobe S' ausgewertet wird
  - Und insgesamt ergibt Frage ⊛ ausgewertet auf S' den gleichen Wert $\frac{b}{10a + 19b}$

D.h.: Auf Grund unserer Stichprobe erhalten wir als Antwort auf Frage (*) den Wert

$$\frac{b}{10a + 19b} \qquad (1)$$

obwohl die tatsächliche Antwort auf dem gesamten Datenstrom der Wert

$$\frac{b}{a+b} \qquad (2)$$

ist.

Konkret für $a = 95$ und $b = 5$ ist

(1) der Wert $\frac{5}{10 \cdot 95 + 19 \cdot 5} = \frac{5}{1045} \approx 0{,}005$

aber (2) der Wert $\frac{5}{95+5} = \frac{5}{100} \approx 0{,}05$

D.h.: Der Wert, den wir auf Grund unserer Stichprobe erhalten, weicht um den Faktor 10 vom tatsächlichen Wert ab!

Bessere Lösung:

Statt jedes einzelne Tupel des Datenstroms mit Wahrscheinlichkeit $\frac{1}{10}$ in die Stichprobe aufzunehmen,
entscheiden wir uns dafür, bei etwa $\frac{1}{10}$ aller Nutzer sämtliche Anfragen des Nutzers in die Stichprobe aufzunehmen
(und alle anderen Nutzer hinsichtlich unserer Stichprobe komplett zu ignorieren).

Für unser Beispiel-Szenario wird dann die Antwort auf Frage (*) über dieser Stichprobe der Wert $\frac{b}{a+b}$ sein – also derselbe Wert, wie wenn wir Frage (*) über dem gesamten Datenstrom beantworten.

Realisierung:
Um diese Stichprobe zu erhalten, nutzen wir eine Hashfunktion

$$h: NID \to \{1, 2, \dots, 10\},$$

wobei NID die Menge aller möglichen Nutzer IDs ist.

Beim Verarbeiten eines Tupels

$t =$ (NutzerID, Anfrage, Zeitpunkt)

berechnen wir h(NutzerID) und nehmen das Tupel t genau dann in unsere Stichprobe auf, wenn h(NutzerID) $= 1$ ist.

Allgemeines Prinzip:

Wir nutzen die Hash-Funktion h hier als "Zufallszahlen-Generator", der die (hier wünschenswerte) Eigenschaft hat, dass er für jeden festen Nutzer immer wieder dieselbe Zahl generiert.

Hinweis:

Falls wir an Stelle einer Stichprobe der Größe $\frac{1}{10}$ eine Stichprobe der Größe $\frac{x}{y}$ für natürliche Zahlen x, y mit $1 \leq x \leq y$ erstellen wollen, können wir eine Hash-Funktion mit Wertebereich $\{1, \dots, y\}$ nutzen und einen Schlüssel K genau dann in die Stichprobe aufnehmen, wenn

$h(K) \leq x$ ist.

(Quellennachweis: Die Darstellung auf den Seiten 21-26 orientiert sich an Kapitel 4.2 "Sampling data in a stream" im Buch "Mining of Massive Data Sets" von Leskovec, Rajamaran, Ullman (Version: 2014))