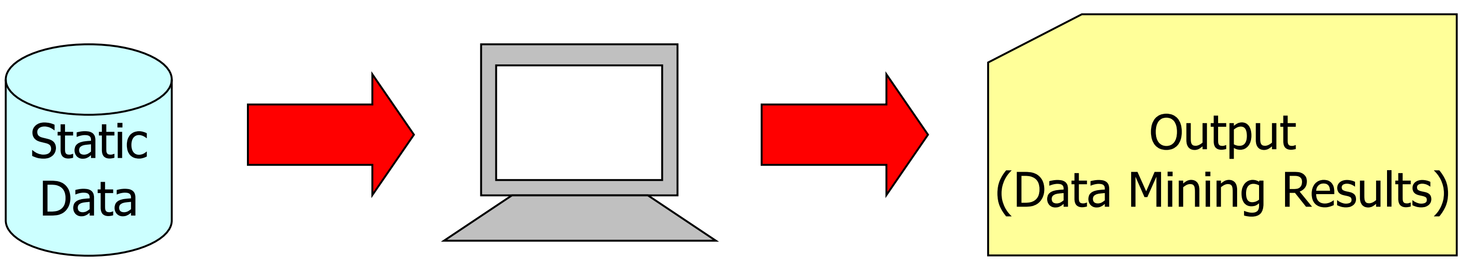
COMP5331

Data Stream

Prepared by Raymond Wong
Presented by Raymond Wong
raywong@cse

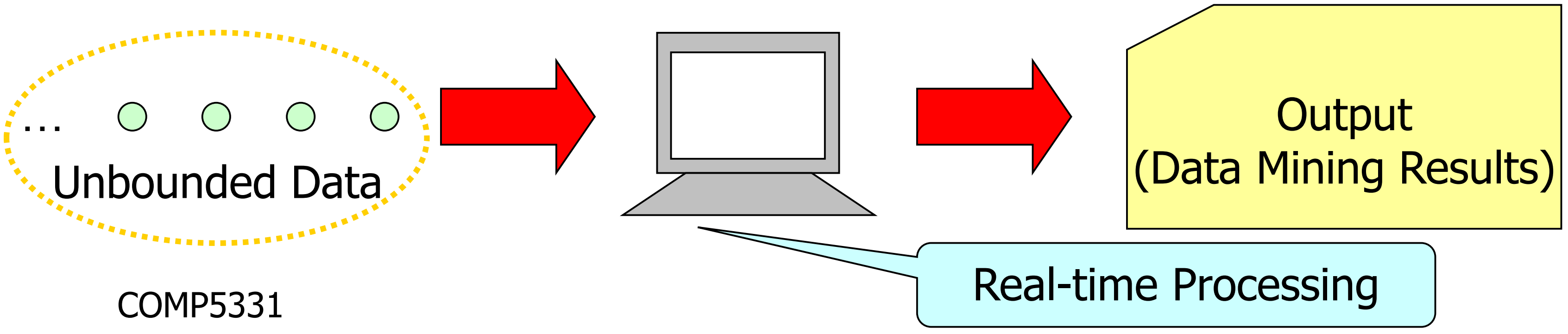
Data Mining over Static Data

1. Association
2. Clustering
3. Classification

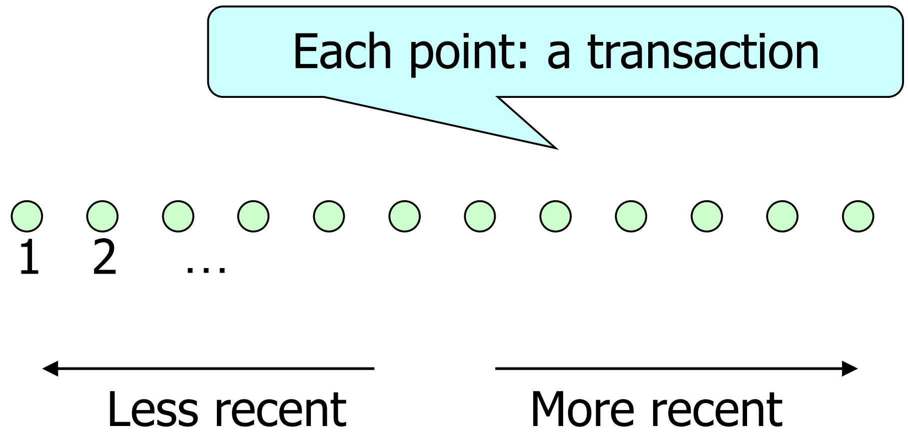


Data Mining over **Data Streams**

1. Association
2. Clustering
3. Classification



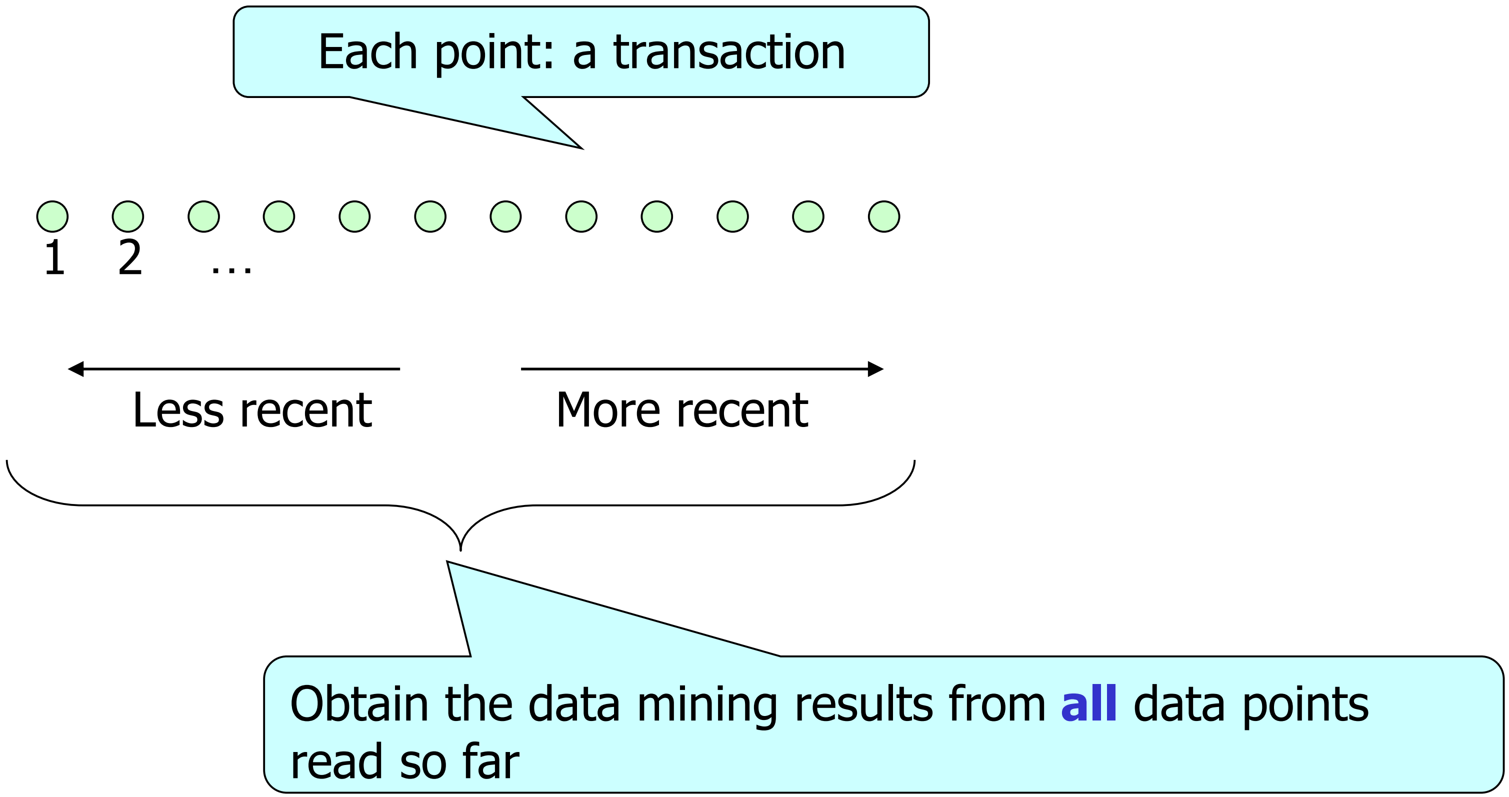
Data Streams



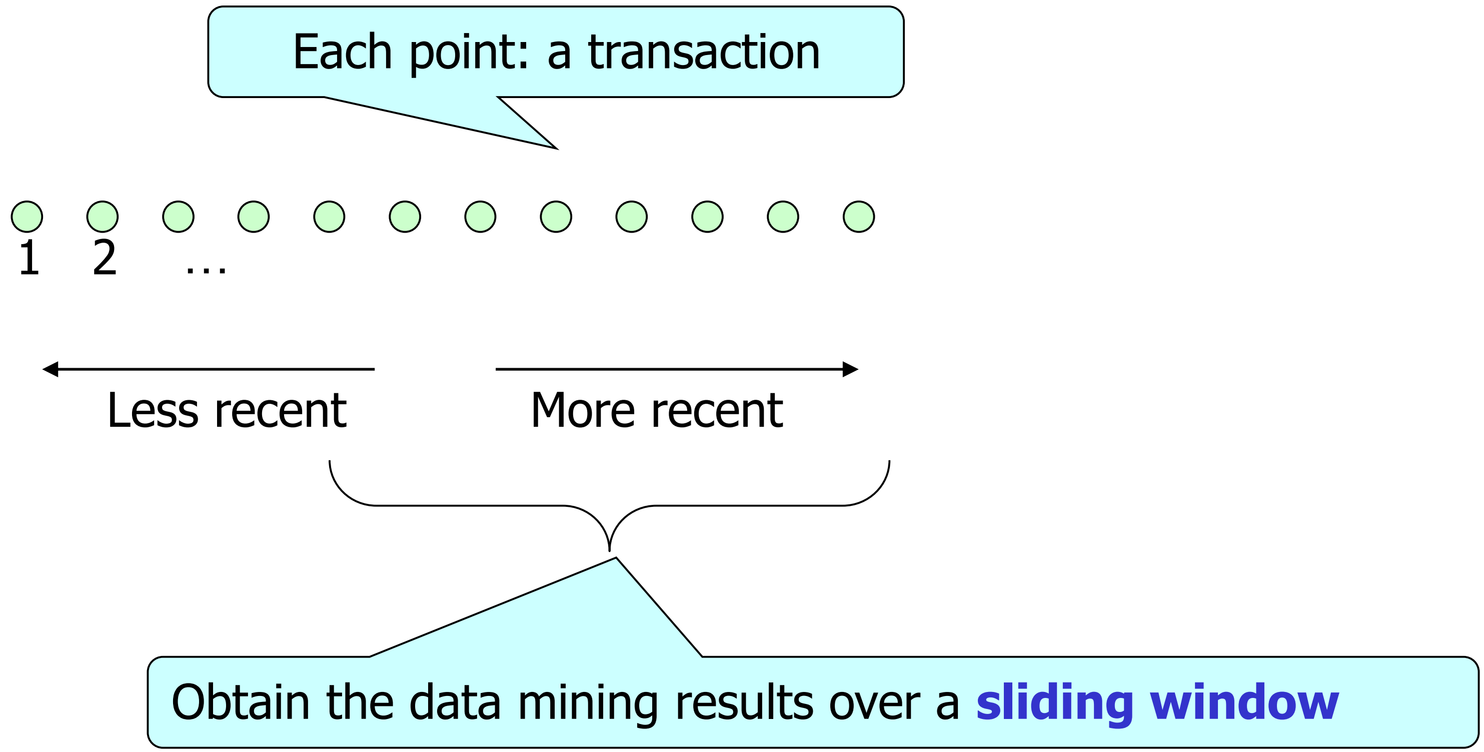
Data Streams

	Traditional Data Mining	**Data Stream Mining**
Data Type	Static Data of Limited Size	Dynamic Data of **Unlimited** Size (which arrives at **high speed**)
Memory	Limited	Limited → More challenging
Efficiency	Time-Consuming	Efficient
Output	Exact Answer	Approximate (or Exact) Answer

Entire Data Streams



Entire Data Streams



Data Streams

- Entire Data Streams
- Data Streams with Sliding Window

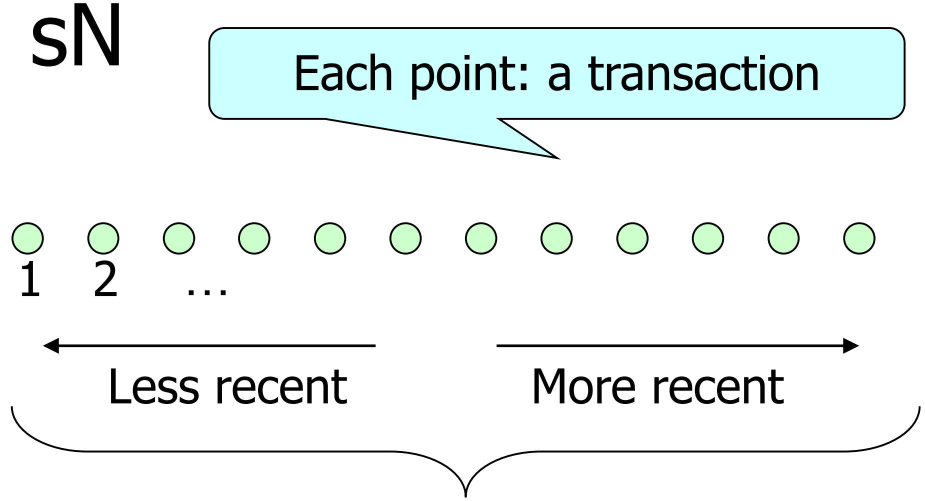
Entire Data Streams

- Association
- Clustering
- Classification

Frequent pattern/item

Frequent Item over Data Streams

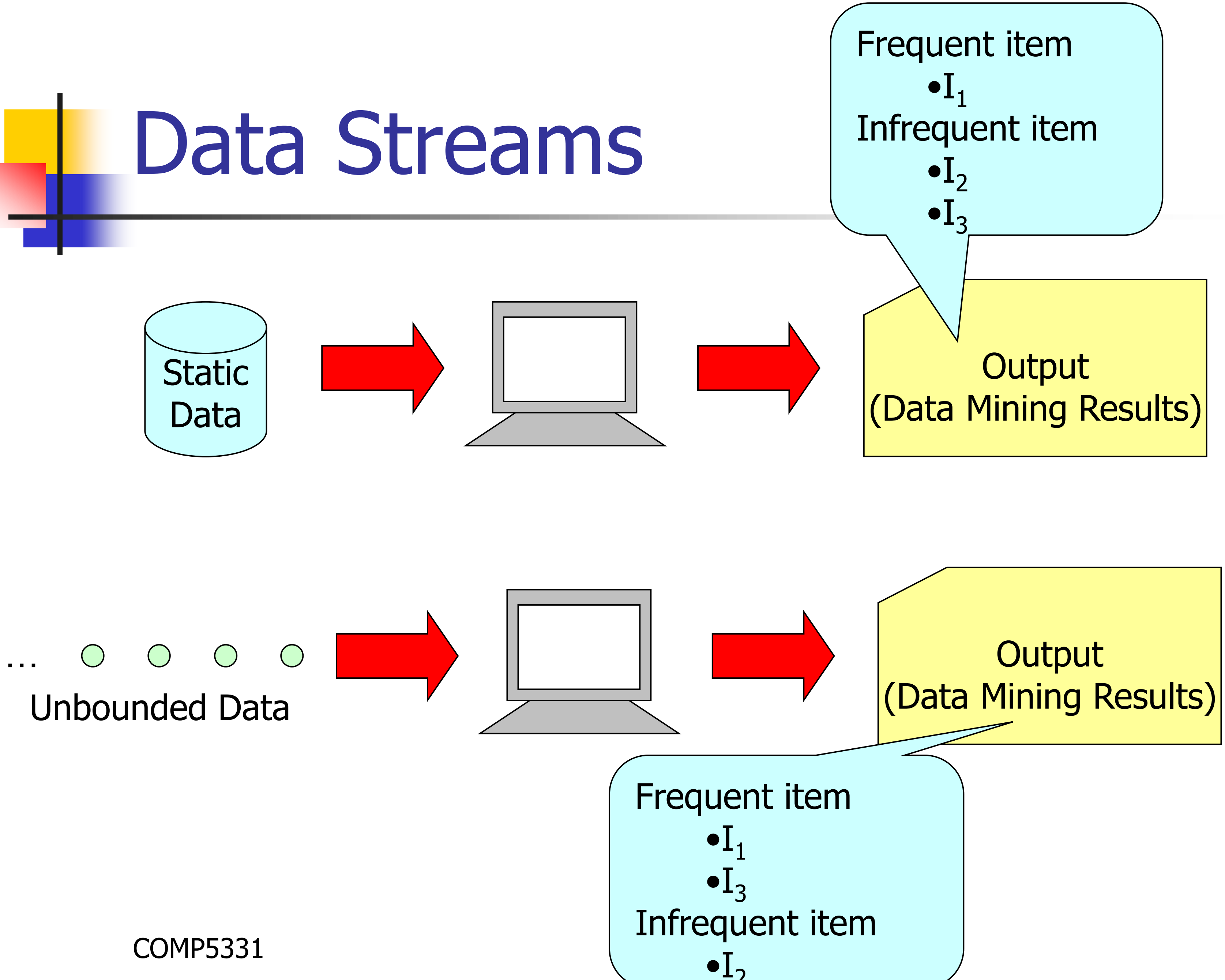
- Let N be the length of the data streams
- Let s be the support threshold (in fraction) (e.g., 20%)
- **Problem:** We want to find all items with frequency >= sN



Data Streams

	Traditional Data Mining	**Data Stream Mining**
Data Type	Static Data of Limited Size	Dynamic Data of **Unlimited** Size (which arrives at **high speed**)
Memory	Limited	Limited → More challenging
Efficiency	Time-Consuming	Efficient
Output	Exact Answer	Approximate (or Exact) Answer

Data Streams



False Positive/Negative

- E.g.
  - Expected Output
    - Frequent item
      - I_1
    - Infrequent item
      - I_2
      - I_3
  - Algorithm Output
    - Frequent item
      - I_1
      - I_3
    - Infrequent item
      - I_2

False Positive
-The item is classified as frequent item
-In fact, the item is infrequent

Which item is one of the false positives? I_3

More? No.

No. of false positives = 1

If we say:
The algorithm has no false positives.

All true infrequent items are classified as infrequent items in the algorithm output.

False Positive/Negative

- E.g.
  - Expected Output
    - Frequent item
      - I_1
      - I_3
    - Infrequent item
      - I_2
  - Algorithm Output
    - Frequent item
      - I_1
    - Infrequent item
      - I_2
      - I_3

False Negative
-The item is classified as infrequent item
-In fact, the item is frequent

Which item is one of the false negatives? I_3

More? No.

No. of false negatives = 1

No. of false positives = 0

If we say:
The algorithm has no false negatives.

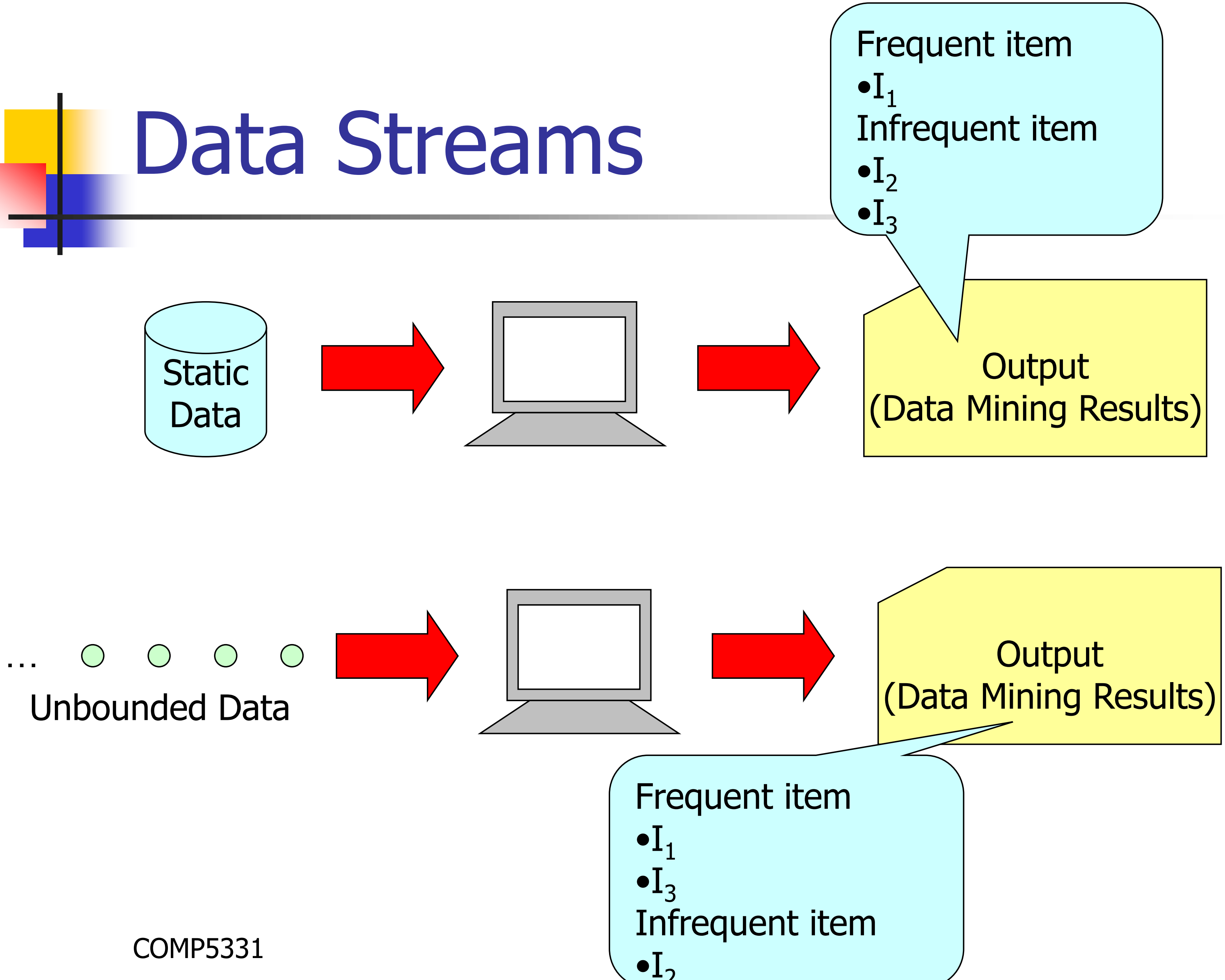
All true frequent items are classified as frequent items in the algorithm output.

Data Streams

	Traditional Data Mining	Data Stream Mining
Data Type	Static Data of Limited Size	Dynamic Data of **Unlimited** Size (which arrives at **high speed**)
Memory	Limited	Limited → More challenging
Efficiency	Time-Consuming	Efficient
Output	Exact Answer	Approximate (or Exact) Answer

We need to introduce an input error parameter ε

Data Streams



Da

Store the statistics of all items
- I_1: 10
- I_2: 8
- I_3: 12

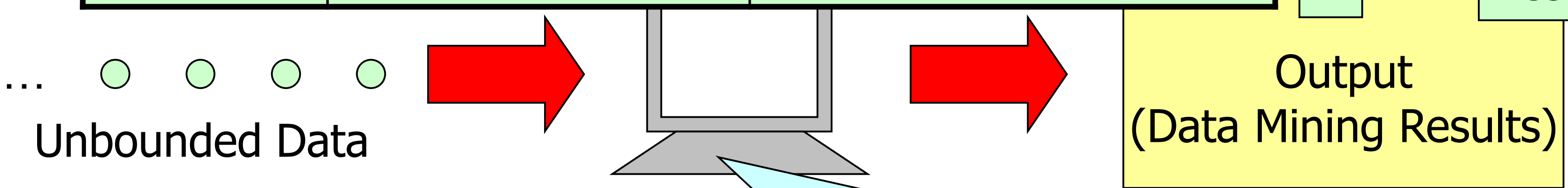
N: total no. of occurrences of items

$N = 20$

$\varepsilon = 0.2$

$\varepsilon N = 4$

Item	True Frequency	Estimated Frequency	Diff. D	D <= εN ?
I_1	10	10	0	Yes
I_2	8	4	4	Yes
I_3	12	10	2	Yes



Estimate the statistics of all items
- I_1: 10
- I_2: 4
- I_3: 10

ε-deficient synopsis

- Let N be the current length of the stream (or total no. of occurrences of items)
- Let ε be an input parameter (a real number from 0 to 1)

- An algorithm maintains an **ε-deficient synopsis** if its output satisfies the following properties
  - **Condition 1:** There is no **false negative**.

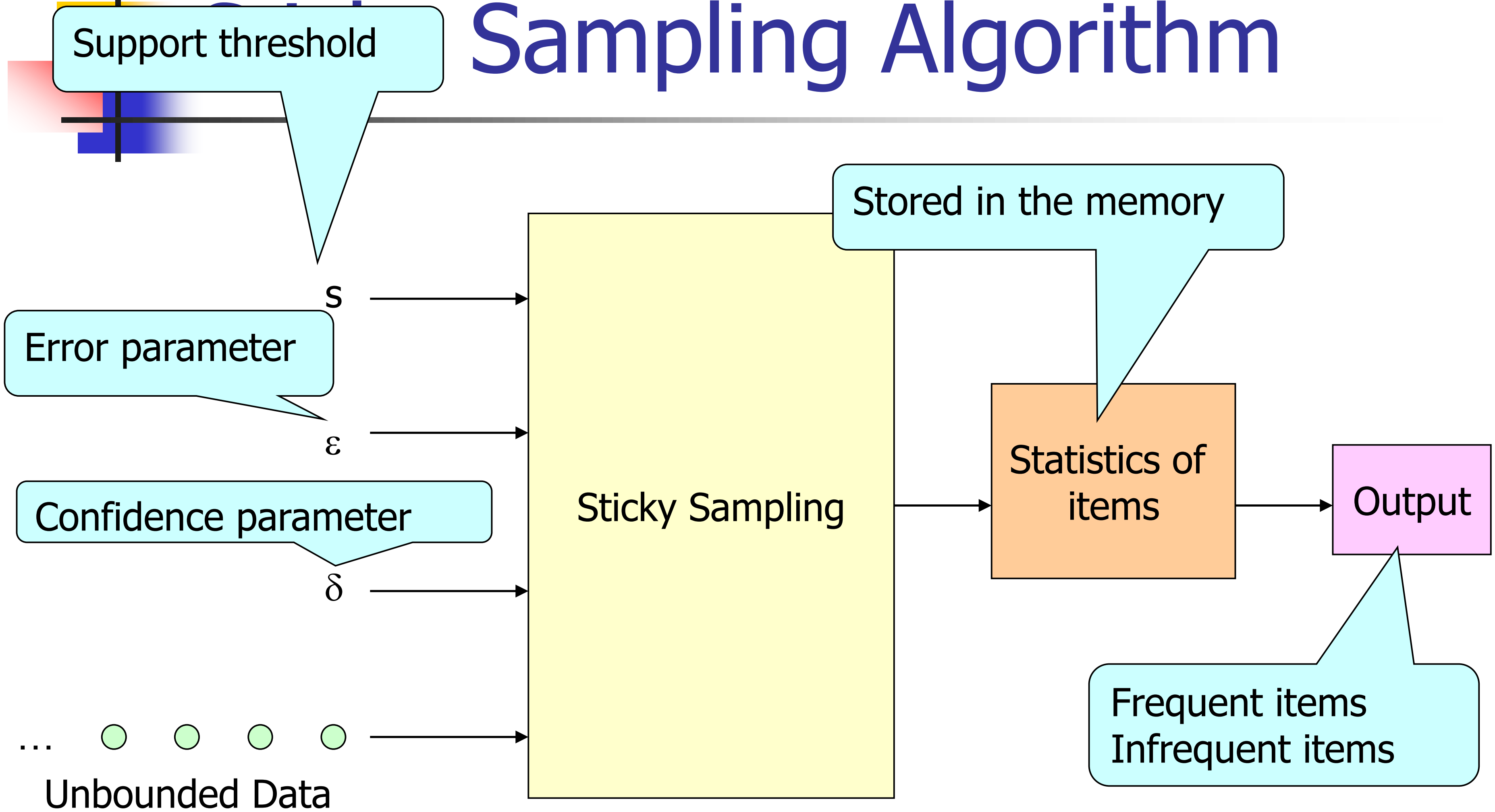
    All true frequent items are classified as frequent items in the algorithm output.

  - **Condition 2:** The difference between the estimated frequency and the true frequency is at most εN.
  - **Condition 3:** All items whose true frequencies less than (s-ε)N are classified as infrequent items in the algorithm output

Frequent Pattern Mining over Entire Data Streams

- Algorithm
  - Sticky Sampling Algorithm
  - Lossy Counting Algorithm
  - Space-Saving Algorithm

Sampling Algorithm



Sticky Sampling Algorithm

- The **sampling rate** r varies over the lifetime of a stream
- Confidence parameter δ (a small real number)
- Let $t = \lceil 1/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$

Data No.	r (sampling rate)
1 ~ 2t	1
2t+1 ~ 4t	2
4t+1 ~ 8t	4
…	…

Sticky S… gorithm

e.g.
s = 0.02
ε = 0.01
δ= 0.1

t = 622

- The **sampli…** …over the lifetime of a stream
- Confidence parameter δ (a small real number)
- Let $t = \lceil 1/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$

1~1244

1245~2488

2489~4976

Data No.	r (sampling rate)
1 ~ 2*622	1
2*622+1 ~ 4*622	2
4*622+1 ~ 8*622	4
…	…

Sticky S… gorithm

e.g.
s = 0.5
ε = 0.35
δ= 0.5

t = 4

- The **sampli…** …over the lifetime of a stream
- Confidence parameter δ (a small real number)
- Let $t = \lceil 1/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$

1~8

9~16

17~32

Data No.	r (sampling rate)
1 ~ 2*4	1
2*4+1 ~ 4*4	2
4*4+1 ~ 8*4	4
…	…

Sticky Sampling Algorithm

element

Estimated frequency

1. S: empty list
   → will contain (e, f)
2. When data e arrives,
   - if e exists in S, increment f in (e, f)
   - if e does not exist in S, add entry (e, 1) with prob. 1/r (where r: sampling rate)
3. Just after r changes,
   - For each entry (e, f),
     - Repeatedly toss a coin with P(head) = 1/r until the outcome of the coin toss is head
     - If the outcome of the toss is tail,
       - Decrement f in (e, f)
     - If f = 0, delete the entry (e, f)
4. **[Output]** Get a list of items where
   $f + \varepsilon N >= sN$

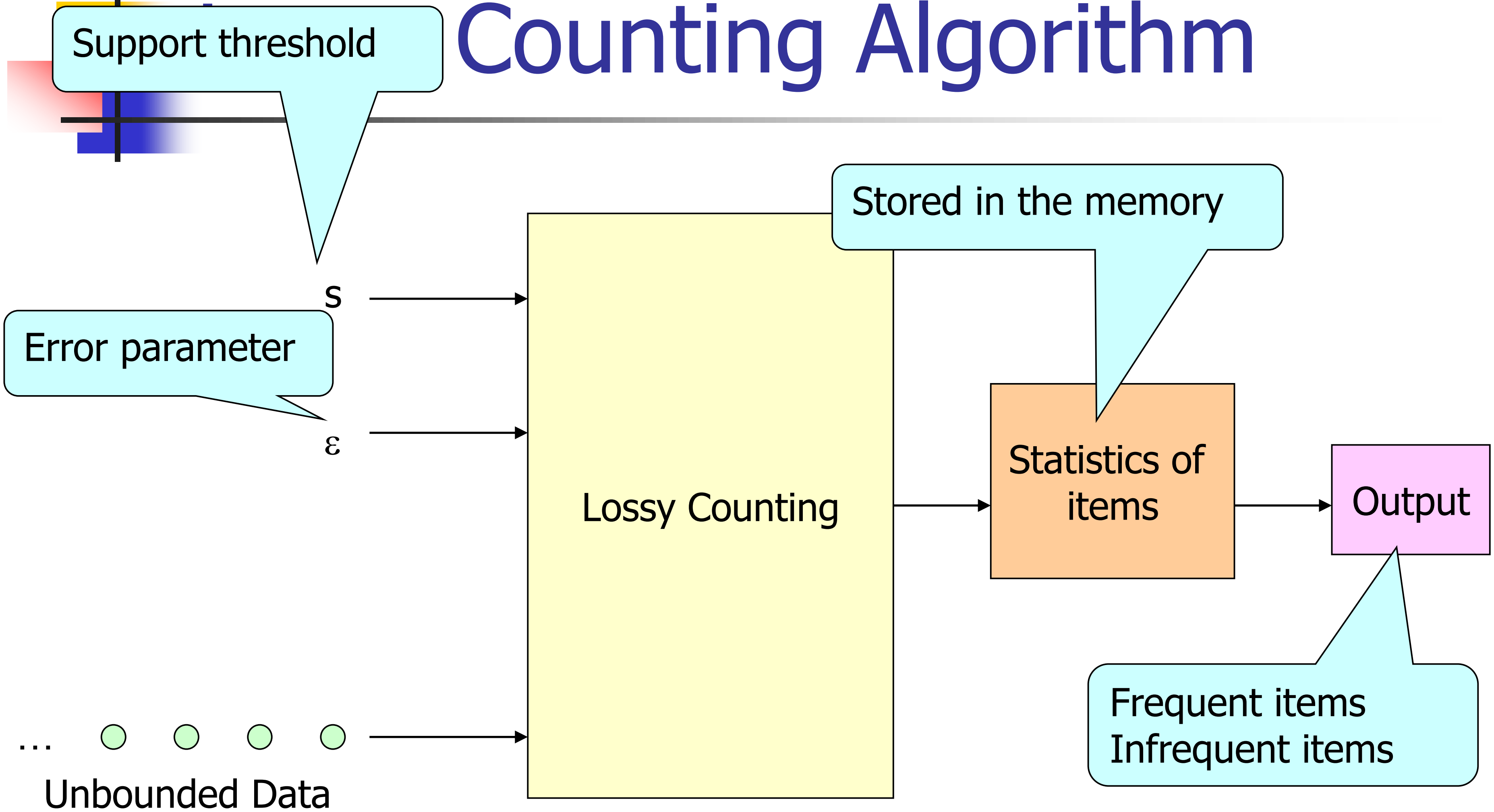
Analysis

- **ε-deficient synopsis**
  - Sticky Sampling computes an ε-deficient synopsis with probability at least $1-\delta$
- **Memory Consumption**
  - Sticky Sampling occupies at most $\lceil 2/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$ entries on average

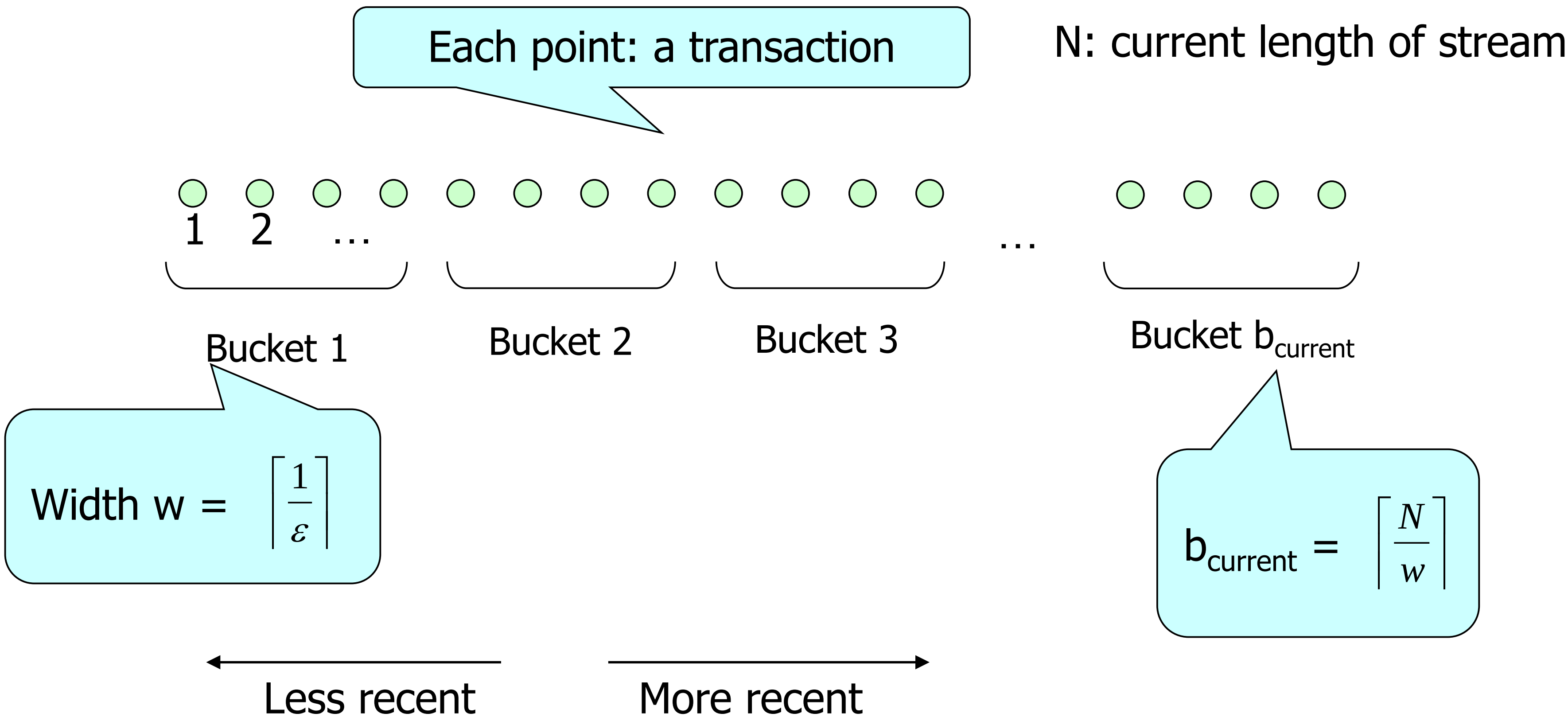
Frequent Pattern Mining over Entire Data Streams

- Algorithm
  - Sticky Sampling Algorithm
  - Lossy Counting Algorithm
  - Space-Saving Algorithm

Counting Algorithm



Lossy Counting Algorithm



Lossy Counting Algorithm

element

Frequency of element since this entry was inserted into D

Max. possible error in f

1. D: Empty set
   - Will contain (e, f, Δ)
2. When data e arrives,
   - If e exists in D,
     - Increment f in (e, f, Δ)
   - If e does not exist in D,
     - Add entry $(e, 1, b_{current}-1)$
3. Remove some entries in D whenever
   $N \equiv 0 \bmod w$
   (i.e., whenever it reaches the bucket boundary)
   The rule of deletion is:
   (e, f, Δ) is deleted if
   $$f + \Delta <= b_{current}$$
4. **[Output]** Get a list of items where
   $$f + \varepsilon N >= sN$$

Lossy Counting Algorithm

- **ε-deficient synopsis**
  - Lossy Counting computes an ε-deficient synopsis
- **Memory Consumption**
  - Lossy Counting occupies at most $\lceil 1/\varepsilon \log(\varepsilon N) \rceil$ entries.

Comparison

e.g.
s = 0.02
ε = 0.01
δ= 0.1
N = 1000

	ε-deficient synopsis	Memory Consumption
Sticky Sampling	1-δ confidence	$\lceil 2/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$
Lossy Counting	100% confidence	$\lceil 1/\varepsilon \log(\varepsilon N) \rceil$

Memory = 1243

Memory = 231

Comparison

e.g.
$s = 0.02$
$\varepsilon = 0.01$
$\delta = 0.1$
$N = 1,000,000$

	ε-deficient synopsis	Memory Consumption
Sticky Sampling	$1-\delta$ confidence	$\lceil 2/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$
Lossy Counting	100% confidence	$\lceil 1/\varepsilon \log(\varepsilon N) \rceil$

Memory = 1243

Memory = 922

Comparison

e.g.
s = 0.02
ε = 0.01
δ= 0.1
N = 1,000,000,000

Memory = 1243

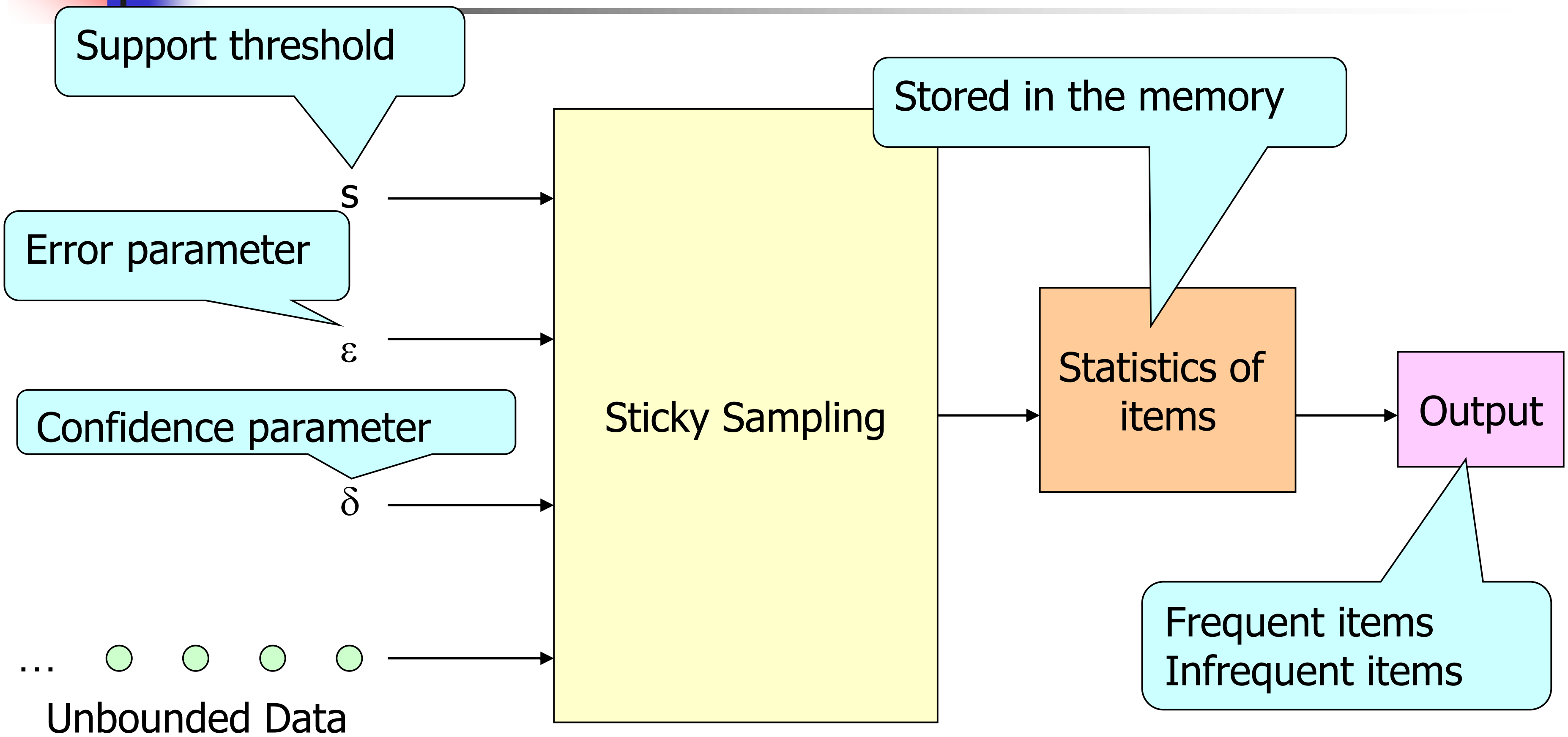
	ε-deficient synopsis	Memory Consumption
Sticky Sampling	1-δ confidence	$\lceil 2/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$
Lossy Counting	100% confidence	$\lceil 1/\varepsilon \log(\varepsilon N) \rceil$

Memory = 1612

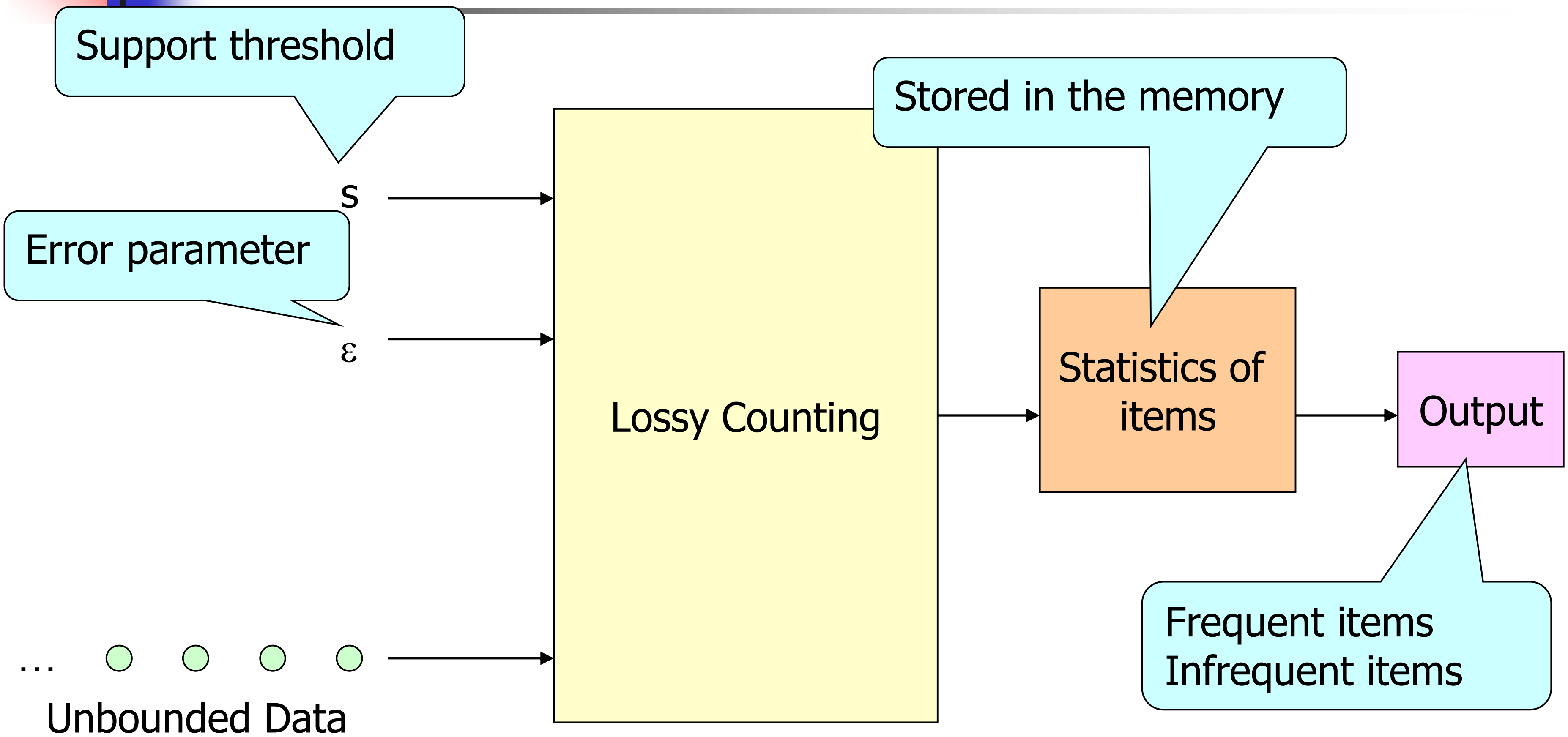
Frequent Pattern Mining over Entire Data Streams

- Algorithm
  - Sticky Sampling Algorithm
  - Lossy Counting Algorithm
  - Space-Saving Algorithm

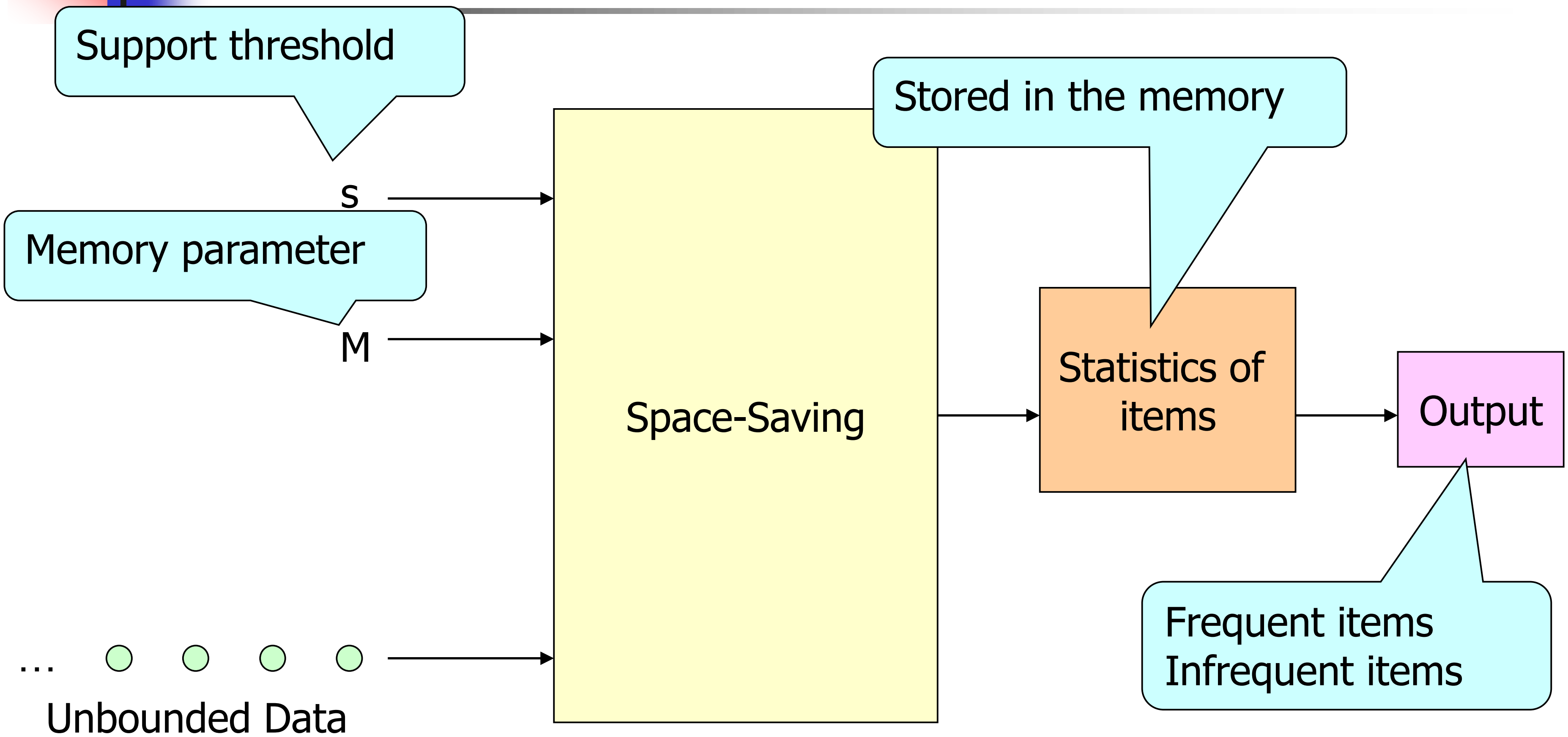
Sticky Sampling Algorithm



Lossy Counting Algorithm



Space-Saving Algorithm



Space-Saving

- M: the greatest number of possible entries stored in the memory

Space-Saving

element

Frequency of element since this entry was inserted into D

Max. possible error in f

1. D: Empty set
   - Will contain (e, f, Δ)
2. $p_e = 0$
3. When data e arrives,
   - If e exists in D,
     - Increment f in (e, f, Δ)
   - If e does not exist in D,
     - If the size of D = M
       - $p_e \leftarrow \min_{e \in D} \{f + \Delta\}$
       - Remove all entries e where $f + \Delta \leq p_e$
     - Add entry $(e, 1, p_e)$
4. **[Output]** Get a list of items where
   $f + \Delta >= sN$

Space-Saving

- **Greatest Error**
  - Let E be the greatest error in any estimated frequency.

$$E \leq 1/M$$

- **ε-deficient synopsis**
  - Space-Saving computes an ε-deficient synopsis if $E \leq \varepsilon$

Comparison

e.g.
$s = 0.02$
$\varepsilon = 0.01$
$\delta = 0.1$
$N = 1{,}000{,}000{,}000$

	ε-deficient synopsis	Memory Consumption
Sticky Sampling	$1-\delta$ confidence	$\lceil 2/\varepsilon \ln(s^{-1}\delta^{-1}) \rceil$
Lossy Counting	100% confidence	$\lceil 1/\varepsilon \log(\varepsilon N) \rceil$
Space-Saving	100% confidence where $E <= \varepsilon$	M

Memory = 1243

Memory = 1612

Memory can be very large (e.g., 4,000,000)
Since $E <= 1/M$
➔ the error is very small

Data Streams

- Entire Data Streams
- Data Streams with Sliding Window

Data Streams with Sliding Window

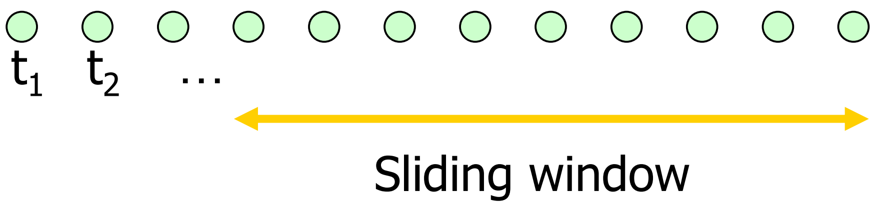
- Association
- Clustering
- Classification

Frequent pattern/itemset

Sliding Window

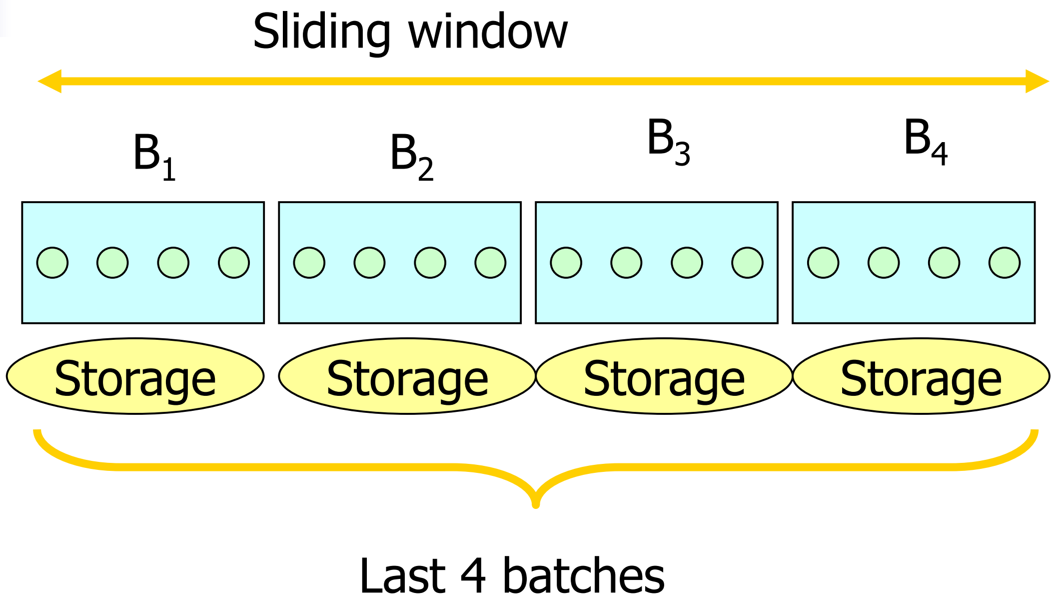
- Mining Frequent Itemsets in a sliding window
- E.g.

  t_1: I_1 I_2
  t_2: I_1 I_3 I_4
  …



- To find frequent itemsets in a sliding window

Sliding Window



Sliding Window