Data Mining & Machine learning (DMML)

Extract insights from data

Extrapolate from past data – Prediction

Data collection → Cleaning

Not focus on this

Patterns in data – Statistics

Analytical – build mathematical relationship

"Equations"

Use computational power to derive relations from volumes of data

Two types of problems

- Patterns
  - Unsupervised
    - Clustering in groups
- Predictions
  - Supervised
    - Use a "model"
      Find values of parameters from historical data

Market-Basket Analysis

Shopping baskets

Identify frequently occurring combinations

Items $I = \{i_1, i_2, \ldots, i_N\}$

Basket = Transaction $t \subseteq I$

$T = \{t_1, t_2, \ldots, t_M\}$

Threshold : - how frequent

is a subset $X \subseteq I$?

$$\frac{\#\ t_i \text{ containing } X}{M} = \text{Support}(X)$$

Frequent: $\text{Support}(X) \geq \text{Threshold}$

↑ Input parameter

Given threshold, find all X s.t. $\text{Support}(X) \geq \text{threshold}$.

Given a list of integers

$y_1, y_2, \ldots, y_M$

find all integers that appear with

support ≥ 0.01

Dictionary

Keys - integers (that appear)

Values - frequency

Additional constraint: All values $\leq 10^6$

Index ≡ Input Integers

Array Count $[0 \ldots 10^6]$

Initialize to 0

See i → increment Count$[i]$

Items : $I = \{i_1, \ldots, i_N\}$

Fix $X \subseteq I$ Go through $T = \{t_1, \ldots, t_n\}$
& count frequency of X

To do this for every $X \subseteq I$ $|I| = N$

Potentially – 2^N subsets

Space – Counters take too much space

Time – Each transaction requires 2^N updates to be checked

Calculation Assume each $t_i \leq 10$ items

$|I| = N = 10^6$

$|T| = M = 10^9$

Threshold is 0.01

Simpler question

How many singleton sets $\{i_k\}$ are frequent?

$|I| = 10^6$

$|T| = 10^9$

Thresh = 0.01

$|t_i| \leq 10$

How many items in T?

See at most 10^{10} items across all of T

Frequent item must appear 10^7 times

Frequent items at most $\frac{10^{10}}{10^7}$

$= 1000$

Suppose $\{x, y\}$ appears frequently

What can we conclude about $\{x\}$, $\{y\}$?

If $\{x\}$ or $\{y\}$ is <u>not</u> frequent,

$\{x, y\}$ <u>cannot</u> be frequent

Example → 1000 potentially frequent $\{x\}$

$\{x, y, 2\}$ frequent → $\{x, y\}$, $\{y, 2\}$, $\{x, 2\}$ all freq.

A Priori Principle

For X to be frequent, every subset of X must be frequent

(layered) A Priori Algorithm

Count frequent sets of size 1

↳ List of candidates of size 2

↳ Count frequent sets of size 2

↳ Candidates of size 3 → count size 3 ...

When do we stop?

- Candidate size exceeds max ti size
- Or the frequent itemset count goes to zero at some level
- Use case – stop at some small level

Given $|t_i| \leq K$

Make upto K passes over T

level 1

level 2

$\simeq 10^9$ ops/sec

⋮

level k

Computational bottleneck

F_i – frequent sets at level i

to C_{i+1} – Candidates at level i+1

$F_i \rightarrow C_{i+1} = \{X$ of size $i+1$, every i-subset of X is in $F_i\}$

Instead of C_{i+1}, can use any D_{i+1} that is a superset of C_{i+1}

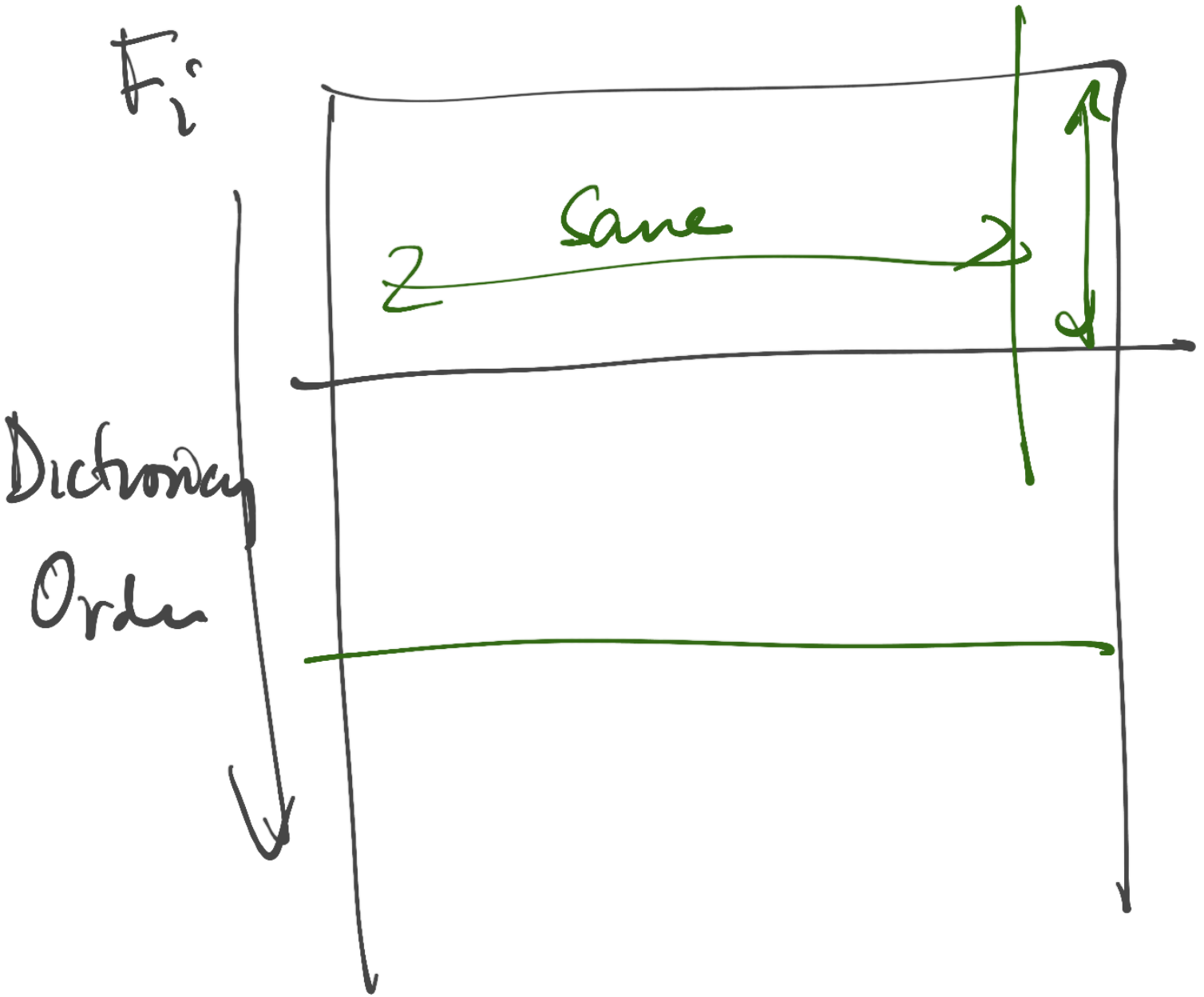
Strategy Order the items $i_1 < i_2 < \dots < i_m$

Enumerate any $t \subseteq I$ in ascending ordering

take two transactions in F_i

$$t_1 = \{j_1, j_2, \dots, j_{l-1}, j_i\}$$
$$t_2 = \{j_1, j_2, \dots, j_{l-1}, j_i'\}$$

$$t_{12} = \{j_1, j_2, \dots, j_{l-1}, j_l, j_i'\}$$


Fi
Same
Dictionary
Order