Introduction to Information Retrieval:-

Jaccard Coefficient:-

Only focus on presence of word (count the word only once).

$$\text{Jaccard}(A,B) = \frac{|A \cap B|}{|A \cup B|}$$ (Intersection on Union)

- Always gives score betn [0-1].

Problems/Issues:-

- Does not consider frequency.
- Rare items/terms should be given higher weightage.

Query Document Method:-

Binary.

Words \ Documents	Doc 1	Doc 2	...
the	0	1	..
a	1	0	..
⋮	⋮	⋮	⋮

Term-document count matrix.

Word \ Document	Doc 1	Doc 2	...
the	0	3	..
a	2	1	..
⋮	⋮	⋮	⋮

Term Frequency:-

Relevance $\propto$ freq.

Log freq. weighting.

$$w_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d} & \text{if } tf \neq 0. \\ 0 \end{cases}$$

$$\therefore \text{Scores} = \sum_{t \in (q \cap d)} (1 + \log_{10} tf_{t,d})$$

* Rare Items are more informative. *

Idf weight (Inverse document frequency)

$$idf_t = \log_{10}\left(\frac{N}{df_t}\right)$$

df_t = No. of documents which contains the term.

eg.

Term	df_t	idf_t
Cal purina	10	5
Animal	100	4
Sunday	1000	3
fly	10000	2
under	100000	1
the	1000000	0

N = 1 million.

Association Rule Mining :- 10 March.

Market-Basket Transactions :-

- Itemset : Collection of one or more items.
- Support Count (σ) : Freq. of occurrence of an itemset.
- Support : Fraction of transactions that contain an itemset
- Frequent Itemset : An itemset whose support is $\geq$ minsup threshold.

Support : (s) for Rule $\{Milk, Diaper\} \rightarrow \{Beer\}$.

$$- s = \frac{\sigma(Milk, Diaper, Beer)}{|T|} = \frac{2}{5} = 0.4.$$

Confidence : (c)

$$- c = \frac{\sigma(Milk, Diaper, Beer)}{\sigma(Milk, Diaper)} = \frac{2}{3} = 0.67.$$

Task :-

- Find support $\geq$ min-sup.
  confidence $\geq$ min-conf.

~ Brute fore (Computationally Expensive).

→ Decouple the support and confidence.

Two-step Approach :-

1. Frequent Itemset Generation.
   - Generate all itemset with support $\geq$ min-sup.
2. Rule Generation.

Now, even Frequent Itemset Generation is very expensive.

~ Brute force : $O(NMw)$.

→ Reduce no. of candidates : M

→ Reduce no. of Transactions : N.

→ Reduce no. of Comparisons : (MN).

Apriori Principle :-

- If an itemset is frequent, then all of it's subset must also be frequent.

i.e. $\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$.

- This is called as anti-monotone property of support.

Apriori Algorithm :- 11 March.

Candidate Generation :- [Consider the order].

- $F_K = F_{K-1} \times F_1$ → To avoid duplicates, take the lexicographical order, and merge only the higher value. ~~subsequence~~.
- $F_K = F_{K-1} \times F_{K-1}$ (where common ~~prefix~~ prefix length is $K-2$)

[Take prefix to eliminate the duplicates].

□ Alternate $F_{k-1} \times F_{k-1}$ Method :-

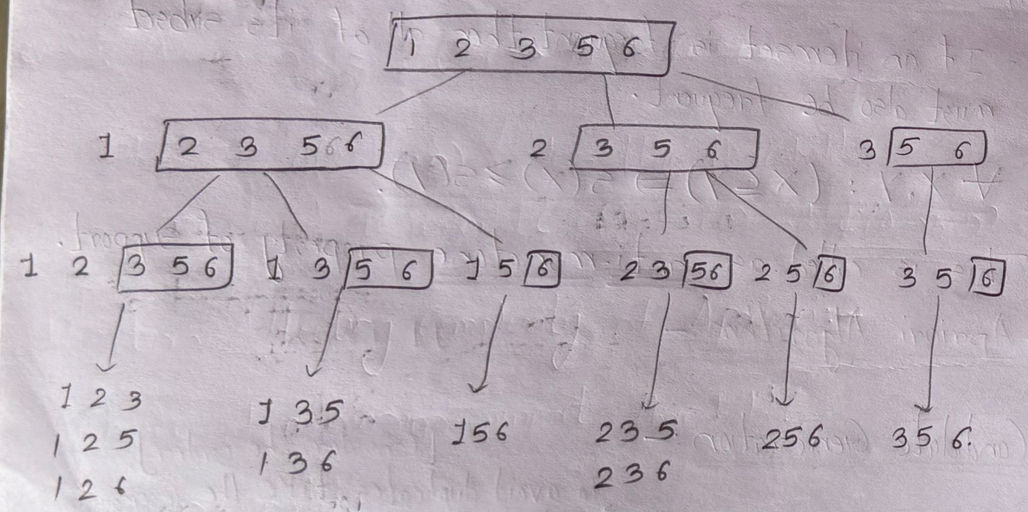
ABC, BCD — Match Last of First and First of Last.

⊛ Now Prune.

- **Algorithm :- Apriori Algo.**
  - (i) Candidate Generation.
  - (ii) Candidate Pruning.
  - (iii) Support Counting.
  - (iv) Candidate Elimination.

Now Counting: By hashing.

eg. Finding all the itemsets present in a transaction.

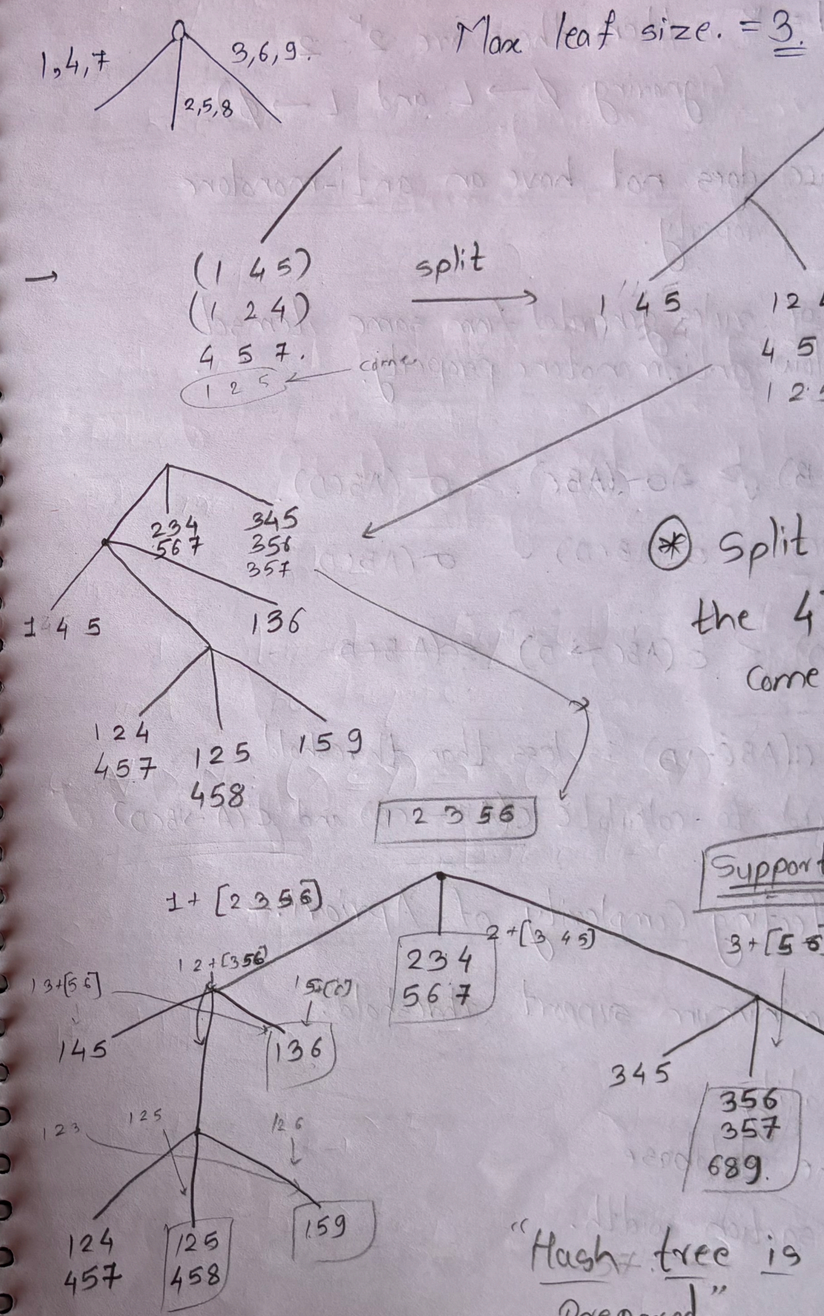


□ Support Counting Using a Hash Tree :-

Hash Function.

15 Candidate itemset of length 3.

Max leaf size. = 3.



⊛ Split, when the 4th element comes. ⊛

Support Counting.

Reducing the search set...

"Hash tree is now prepared."

search reduced from 15 to 9

□ Rule Generation :-

If $|L| = K$, then there are $2^K - 2$ (ignoring $\phi \rightarrow L$ and $L \rightarrow \phi$).

* Confidence does not have an anti-monotone property.

* Confidence of rules generated from same itemset does follow anti-monotone property.

$$\frac{\sigma(A)}{\sigma(ABCD)} \geq \frac{\sigma(AB)}{\sigma(ABCD)} \geq \frac{\sigma(ABC)}{\sigma(ABCD)} \geq \frac{\sigma(ABCD)}{\sigma(ABCD}$$

} inverse.

$$c(A \rightarrow BCD) \leq c(AB \rightarrow CD) \leq c(ABC \rightarrow D) \cancel{\leq c(ABCD}$$

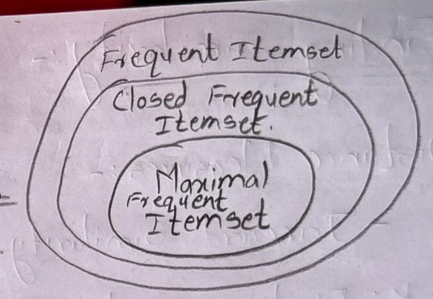
Thus if, $c(ABC \rightarrow D)$ is less than threshold, no need to calculate $c(AB \rightarrow CD)$ and $c(A \rightarrow BCD)$

□ Factors Affecting Complexity of Apriori:-

- Choice of minimum support threshold.
- Dimensionality.
- Size of the database.
- Average transaction width.

• **Maximal Frequent Itemset:-**

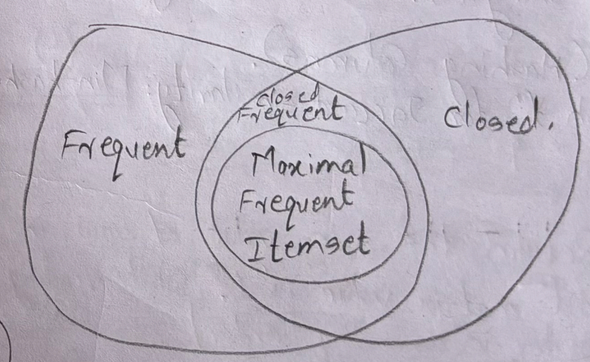
Itemset which is frequent, and none of whose immediate superset is frequent.



• **Closed Frequent Itemset:-**

Itemset which is frequent, and none of it's immediate superset has the same support count as the itemset.

Also, X is not closed if atleast one of it's immediate superset has same support count as X.



* Q. One solved problem on (on Assignment Paper).
- Entropy
- Apriori Principle.

□ **Finding Similarity** :- [J. Leskovec, A. Rajaraman, J. Ullman
Mining of Massive Datasets
www.mmds.org]

- Scene completion can also be one application.

{PPTs for this}

◇ **Distance Measures** :-

- Jaccard Similarity.

• **Task: Finding Similar Documents** :-

Steps :-

~ Shingling (Convert document to sets).
~ Min-Hashing. (Convert large sets to short signatures).
~ Locality-Sensitive Hashing. (LSH)
(Focus on pairs of signature).

□ **Shingling** :-

- k-shingle (or k-gram). ($k=5$ for smaller docs, $k=10$ for larger docs.)
- Compressing shingles.
- Signatures (Hashing Columns).

Suitable hash f^n for Jaccard similarity: Min-hashing.

□ **Min-hashing** :-

Min-hashing :- Return the min value of row, in which the Input matrix value is "1". [while reading the permutation order in ascending order].

π = Random Permutation:

$h_\pi(c)$ = f^n which returns the index of 'first' (in permuted order of π) in which the column value is '1'.

⊛ $Pr[h_\pi(c_1) = h_\pi(c_2)] = sim(c_1, c_2)$. ⊛

- The similarity in hash (in similarity), is the fraction of indices in which they agree.

□ **Implementation Trick** :- [Single Pass Min-Hash Calculation].

TMP Q in Exam

□ **LSH: First Cut** :-

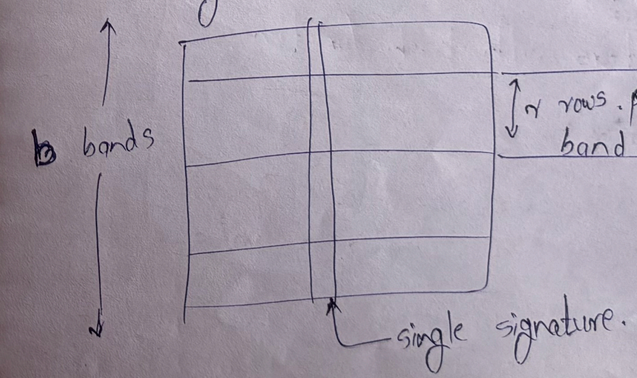
• For Min-Hash Matrix :-

~ Hash the column's of Min-Hash Matrix.

~~~ If ~~~

~ All the Columns lying in the same bucket are group. Candidate Pairs

Signature Matrix M.



Diff^nt bukkets for each band.
~~~

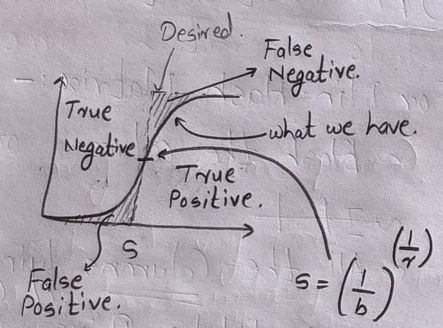
Candidate column pair are those that hash to the same bucket for ≥ 1 band.

[*] $1-(1-s^r)^b$ is high.

Thus, there is very high chance that you will not miss a candidate pair is very high.
(False-negative is reduced).

$y = 1-(1-s^r)^b$ for $r=5$; $b=20$.

s	y
0.1	0.0002
0.2	0.006381
0.3	0.047494
0.4	0.18605
0.5	0.470051
0.6	0.801902
0.7	0.974781
0.8	0.999644
0.9	1
1.0	1



$s = \left(\frac{1}{b}\right)^{\left(\frac{1}{r}\right)}$

↑ False Positive.
← Threshold
↓ True Positive

□ Edit Distance :- [Minimizing Algo. by DP].

- No. of operations required to convert from source string to destination tree.

remove from src.

T	5	6	7	8	9	8	$\boxed{7}$
E	4	5	6	7	8	7	8
R	3	4	5	6	7	8	9
Q	2	3	4	5	6	7	8
P	1	2	3	4	5	6	7
#	0	1	2	3	4	5	6
	#	A	B	C	D	E	T

Source

Destination.

→ Insert in Dest.

Substitute.

$D(m,n)$

Cost to convert 'm' characters from source to 'n' characters in Destination.

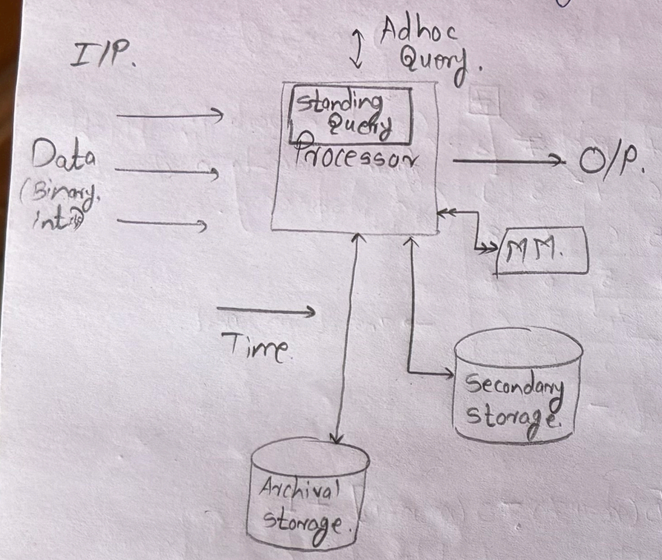
$D(m,n) = \min[D(m-1,n), D(m,n-1)] + 1$.

if $a[i] == b[j]$:

$D(m,n) = \min[D(m,n), D(m-1,n-1)]$

Data Streams :-

[Data Stream Management System (DSMS) :-]



- **Adhoc Queries** :- Required rarely. (One time queries).
- **Standing Queries** :- Executed each and every time.
  will maintain a suitable data, updating each time, so that it can be used to answer Adhoc Queries.

Filtering :-

~ Bloom Filter :-

- Array of filters with hash functions.

eg.

	[Bits in odd pos.] $h_1(x)$ Mod 11.	[Bits in even pos.] $h_2(x)$.	0	1	2	3	4	5	6	7	8	9	10
			–	–	–	–	–	–	–	–	–	–	–
25 → 11001	5	2	0	0	1	0	0	1	0	0	0	0	0
159 → 100 11111	7	0	1	0	1	0	0	1	0	1	0	0	0
585 → 1001001001	9	7	1	0	1	0	0	1	0	1	0	1	0
110110	3	5	1	0	1	1	0	1	0	1	0	1	0

It has False Positive, but cannot have False Negative. [Bloom filters]

- **Click Mining** :- Broken Link.
- **Query Mining** :- Trending Topic.

Sliding Window Concept :-

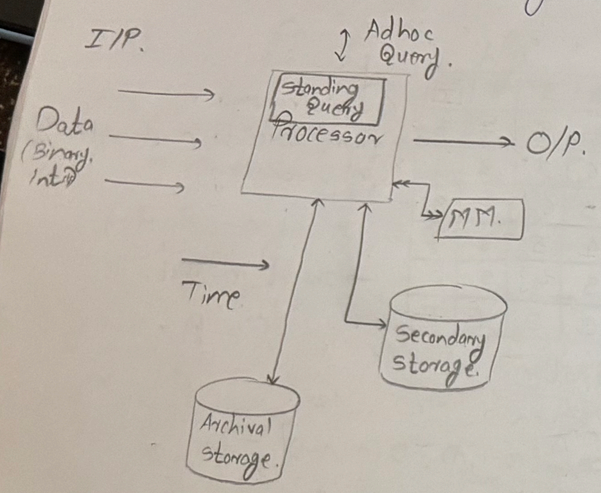
- $$\text{New Avg.} = \text{Old Avg.} + \frac{i - j}{N}$$

  (i → new no., j → old no.)

- **Counting Distinct One's** :-

Data Streams :-

[Data Stream Management System (DSMS):-]



- **Adhoc Queries:-** Required rarely. (One time queries).
- **Standing Queries:-** Executed each and every time.
  Will maintain a suitable data, updating each time, so that it can be used to answer Adhoc Queries.

◇ **Filtering:-**

~ **Bloom Filter:-**

- Array of filters with hash functions.

eg.

	[Bits in odd pos.] $h_1(x)$ Mod 11.	[Bits in even pos.] $h_2(x)$.	0	1	2	3	4	5	6	7	8	9	10
			–	–	–	–	–	–	–	–	–	–	–
25 → 11001	5	2	0	0	1	0	0	1	0	0	0	0	0
159 → 100 1111	7	0	1	0	1	0	0	1	0	1	0	0	0
245 → 100 0100	9	7	1	0	1	0	0	1	0	1	0	1	0
9 → 1001	3	5	1	0	1	1	0	1	0	1	0	1	0

It has False Positive, but cannot have False Negative. [Bloom filters]

- **Click Mining:-** Broken Link.
- **Query Mining:-** Trending Topic.

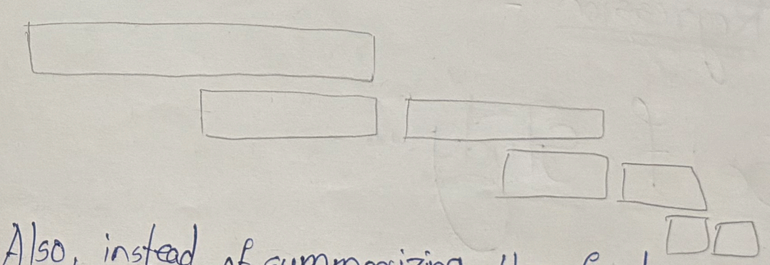
◇ **Sliding Window Concept:-**

- $$\boxed{\text{New Avg.} = \text{Old Avg.} + \frac{i - j}{N}}$$

  (i → new no., j → oldest no.)

- **Counting Distinct One's :-**

- **Counting Bits :-**
  - Obvious Sol^n: keep track of last N bits.
  - Keep track of exponentially increasing sizes of block. (Max 2 of same size).
  - Drop small blocks if they begin at the same point as a larger region start.
  - We can approximate with this approach.



- Also, instead of summarizing the fixed size of boxes, summarize the fixed no. of one's.

Regression :-

- Supervised learning.
- Find the best fitting line / curve.
- Hypothesis : $h(x) = \theta_0 + \theta_1(x)$.
- Minimize the error.
- Cost f^n : $\boxed{J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=0}^{m-1} (h_\theta(x) - y)^2}$
- Contours :- 3D diagram plotted on 2D by slicing in z-plane.
- Gradient Descent Algo :-

$$\boxed{\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad [\text{for } j=0 \text{ and } j=1].}$$

- Do simultaneous update :-

$$\boxed{\begin{aligned} temp0 &= \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) \\ temp1 &= \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) \\ \theta_0 &= temp0 \\ \theta_1 &= temp1. \end{aligned}}$$

$= \theta_0$
$= \theta_1$ } → Incorrect.

Don't update one by one :-

Logistic Regression :-

- Self-Study ...