Data-Mining-Module-5-Important-Topics-PYQs

For more notes visit

https://rtpnotes.vercel.app

1. Describe any two-text retrieval indexing techniques.

What is Text Retrieval?

Imagine you have **a lot of documents**, and someone asks you:

> “Which documents contain the word *banana*?”

To answer this quickly, you need a system that can **find documents fast** — without opening each one manually.

That system is called a **text retrieval indexing technique**.

There are many techniques, but the most popular two are:

1. Inverted Index (Like a word-to-document dictionary)

Imagine this:
You have 3 documents:

- **D1**: "apple banana"

- **D2**: "banana juice"
- **D3**: "apple juice banana"
  Now, you create a list where each **word points to the documents** that contain it:

```
apple  → [D1, D3]
banana → [D1, D2, D3]
juice  → [D2, D3]
```

- This list is called an **inverted index**.
- You just "invert" the normal idea (document → words) and make it (word → documents).

It uses two tables:

- **Term Table**: each term → list of documents (called a posting list)
- **Document Table**: each document → list of terms (optional)

Advantages:

- Fast searching
- Simple to implement
- Used in real search engines (like Google)

Drawbacks:

- Needs more storage (lists can be long)
- Can't handle **synonyms** (e.g., "car" and "automobile") or **polysemy** (e.g., "bank" = river or money)

2. Signature Files (Like a digital fingerprint for each doc)

This is like giving each document a **digital fingerprint**.
You convert each document into a **bit string** (like `10101000`) that represents which words it has.

- Each word is converted into a set of bits using a method called **hashing**
- If a word is in the document, some bits are turned ON (`1`)
- The result is a compact **signature** for that document

Then, when you search for a word or set of words

- You convert the query into a bit string (query signature)
- Compare it with all document signatures
- If it matches, **fetch and check** the document to confirm

Why fetch and check?

Because two different words might activate the same bits (due to multiple-to-one mapping), some documents may look like a match even if they aren't. This is called a **false positive**, and that's why checking is needed.

Advantages:

- Uses **less space** (fixed-size signatures)
- Faster to scan than reading full documents

Drawbacks:

- **False matches** due to overlapping bits
- Need extra checking
- More complex if not filtered properly

2. Compare and contrast the focused crawling and regular crawling techniques.

Introduction to Crawlers

Imagine you want to collect **customer reviews** from various shopping websites.
You could:

- Start at Amazon
- Go to the product page
- Open every review page
- Collect text, rating, and date
- Then go to Flipkart, do the same
- Repeat this for every site

This **automated program** that visits pages, follows links, and gathers data is called a **web crawler** (or spider or robot). Crawlers are the **backbone of search engines**, helping build and update their page indexes.

Regular Crawlers (a.k.a. General or Periodic Crawlers)

What they do:

- Start with some initial links (called **seed URLs**)
- Visit every page linked from them
- Then follow links from those pages
- Keep visiting and collecting data — **regardless of topic**

Use:

- Used by general search engines (like Google, Bing)
- Goal: **Cover as many pages as possible**

Types:

- **Periodic Crawlers**: Crawl the entire web again after a fixed time to update the index.
- **Incremental Crawlers**: Only update parts of the index that have changed recently.

Focused Crawlers (Topic-Specific Crawlers)

What they do:

- Also start with seed URLs
- BUT: Only follow links that seem **relevant to a specific topic**
- If a page is off-topic, it is **ignored along with all its links**

Example:

If the crawler is focused on “customer reviews,” it **won’t waste time** going into sports news or recipe blogs.

How they work:

- Use a **Hypertext Classifier** to score how relevant a page is to the topic
- A **Distiller** finds “hub pages” (pages that link to many useful pages)
- The **Crawler** itself only follows links from high-scoring pages

Feature	Regular Crawlers	Focused Crawlers
Purpose	Crawl the **entire web**, regardless of topic	Crawl pages **related to a specific topic**
Link Following	Follows **all links**	Follows only links from **relevant pages**
Efficiency	Low (visits many unrelated pages)	High (skips irrelevant content)
Precision	Low (broad coverage, less accuracy)	High (topic-specific, accurate)
Components	Basic crawler logic and queue	Uses **classifier**, **distiller**, and crawler
Scalability	Less scalable as web grows	More scalable due to targeted crawling
Used For	Search engines (Google, Bing)	Research, academic studies, topic monitoring

3. Describe the following activities involved in the web usage mining i) Pre-processing activity ii) pattern analysis

Web Usage Mining is the process of extracting useful information from logs of web access, like user click streams (the sequence of clicks a user makes on a website).
Where does the data come from?

- Web server logs
- Application server logs
  What does it do?
- Tracks users' browsing history (which pages they visit and in what order).
- Finds patterns in user behavior (like frequently visited pages, search trends, and associations).
- Helps predict what users might be searching for on the Internet.

i) Pre-processing Activity

Pre-processing is the initial and essential step in Web Usage Mining where raw web log data is cleansed, structured, and transformed into a format suitable for mining.

Key Steps in Pre-processing:

1. **Cleansing:**
   Removes irrelevant or unnecessary data like image requests (`.gif`, `.jpg`, etc.) and bot accesses.
2. **User Identification:**
   1. Determines the unique users from IP addresses, user-agents, or cookies.
   2. Challenge: Multiple users might share an IP (via proxy), or one user may use multiple devices.
3. **Session Identification:**
   1. Groups a sequence of page visits into a single session using timeouts or login-based tracking.
   2. A **session** = a logical unit of user activity (e.g., visiting pages A → B → C).
4. **Path Completion:**
   1. Attempts to fill in missing pages that were actually visited but not recorded (e.g., due to caching).
   2. Example: If log shows A → C and no direct link exists, B might be inferred in between.
5. **Formatting:**
   1. Transforms and structures the cleaned data (e.g., converting page names to unique IDs) to reduce size and improve processing speed.

ii)Pattern Analysis

Pattern analysis is the process of interpreting the mined patterns (from logs) to discover **meaningful, actionable insights**.

Key Activities in Pattern Analysis:

1. **Filtering Meaningless Patterns:**
   1. Removes uninteresting or redundant patterns that do not contribute useful insights.
2. **Comparative Analysis:**
   1. Compares browsing patterns between different user groups (e.g., customers vs. non-customers).
   2. Example: Customers may follow a path like *Home* → *Product* → *Checkout*, while non-customers may stop at *Product*.
3. **Use of g-Sequences:**
   1. Patterns may include wildcards:
      1. `b * c` means: visited page b, followed by any number of pages, then page c.

   2. Helps identify **non-contiguous but frequent patterns**.

4. **Concept Abstraction:**
   Uses concept hierarchies to generalize page content (e.g., *"Laptop"* → *"Electronics"*) for more meaningful patterns.
5. **Similarity Rules:**
   Two patterns are considered **similar** if they share at least the first `n` pages (defined by the user).

4. Differentiate between web content mining and web structure mining.

Feature	Web Content Mining	Web Structure Mining
Definition	Extracts useful data from the actual **content** of web pages.	Analyzes the **structure and links** between or within web pages.
Focus	Focuses on **what is in the page** (text, images, videos, etc.).	Focuses on **how pages are connected** via hyperlinks.
Data Types Mined	Text, images, audio, video, structured data (e.g., product listings).	Links, URL structures, HTML tags (like `<a>`, `<iframe>`), graph connections.
Types	1. Web Page Content Mining 2. Search Result Mining	1. Intrapage Structure Mining (within a page) 2. Interpage Structure Mining (between pages)
Example	Extracting product info from Amazon pages.	Google PageRank analyzing links between pages.
Applications	- Content summarization - Sentiment analysis - Search result enhancement	- SEO ranking - Detecting spam link networks - Improving site navigation
Tools/Techniques	NLP, web scrapers, image processing	Graph theory, link analysis algorithms (like PageRank, HITS)

5. Compare web structure mining and web usage mining.

Feature	Web Structure Mining	Web Usage Mining
Definition	Analyzes the **structure and hyperlink connections** between web pages.	Extracts patterns from **web access logs** that record user browsing behavior.
Focus	Focuses on **how pages are linked** (site architecture, internal/external links).	Focuses on **how users interact** with a website (clickstreams, sessions).
Data Source	HTML/XML code, hyperlinks (from web pages).	Web server logs, browser logs, application server logs.
Goal	Understand the **layout and interconnection** of web content.	Discover **user behavior patterns** and preferences.
Techniques Used	Graph theory, link analysis (PageRank, HITS).	Data mining, pattern recognition, clustering, sequence mining.
Example	Google using PageRank to rank pages based on backlinks.	Amazon recommending products based on browsing and buying history.
Applications	- SEO optimization - Website structure improvement - Spam detection	- Personalized recommendations - UX design improvement - Targeted advertising
Types	1. Intrapage structure 2. Interpage structure	1. Preprocessing 2. Pattern discovery 3. Pattern analysis

6. Explain HITS algorithm with an example.

The HITS (Hyperlink-Induced Topic Search) algorithm is a way to find important web pages, but it looks at the **structure** of the links between pages. It does this by categorizing pages into two types: **hubs** and **authorities**.

What are Hubs and Authorities?

- **Hubs** are web pages that link to many other pages. They're like "directories" that point to useful content. Think of a hub like a page that links to a lot of articles, resources, or useful websites.
- **Authorities** are web pages that are linked to by many other pages. They're like "experts" or "important pages" on a particular topic. These are pages that people tend to link to because

they contain valuable content.

How does HITS work?

1. **Finding relevant pages**:
    1. Imagine you're searching for information about "best laptops." The search engine will find a small group of pages related to laptops (called the **root set**). These are pages that directly match your query.
2. **Expanding the set**:
    1. Once we have this small group of pages, we expand it by adding more pages that link to or are linked by these root pages.
    2. This becomes the **base set**. So now, we have not just the pages you were looking for, but also other pages that are connected through links.
3. **Looking at the connections**:
    1. The HITS algorithm looks at the links between these pages in the base set. If a page is linked to by many other pages, it might be an authority. If a page links to many important pages, it might be a hub.
4. **Calculating hub and authority scores**:
    1. A **hub** score increases if the page links to many authoritative pages.
    2. An **authority** score increases if many hubs link to it.
    3. This process repeats until we find the pages with the highest hub and authority scores.
    4. These are the most important and relevant pages for your search.

Example:

Let’s say you search for “best laptops.”

- You might find a page that links to several reviews, product details, and comparison sites. This page would be a **hub** because it points to many useful resources.
- Then, there might be a page that has a detailed review of the top laptops, and many other pages link to it. This page would be an **authority** because it is the expert or highly recommended by other sources.

The HITS algorithm helps find these types of pages to make sure the results you see are from reputable sources (authorities) and useful directories (hubs)

7. Describe different Text retrieval methods. Explain the relationship between text mining, information retrieval and information extraction.

What is Text Mining?

Text mining is like teaching a computer to **read, understand, and find useful info** from lots of text documents.

Imagine you have thousands of news articles or research papers. You can't read them all, right? Text mining helps you:

- Extract **important information** (e.g., names, places, topics)
- Find **patterns** (e.g., what are people talking about most this year?)
- Discover **hidden insights** (e.g., how a product is being reviewed across websites)
- **Example**: From a set of 10,000 reviews, text mining can find that people love the "battery life" of a phone but often complain about the "camera".

What is Information Retrieval (IR)?

IR is about **searching** through a large collection of documents and **retrieving only the relevant ones** based on a user's query (search).

- Like Google Search. You type "top movies 2024" → it shows only those relevant pages.
- IR is used to:
  - **Locate documents** that match a keyword
  - **Rank documents** based on how relevant they are to the query
- It's **pull-based**: You pull information when you search.

What is Information Extraction (IE)?

IE is a part of text mining. Once you have the documents, IE is used to **pull specific pieces of data** from them.

- For example: From a job description, IE pulls out:
  - Job title
  - Skills required
  - Salary

    - Location

Concept	Description	Relation
Text Mining	Extract patterns and insights from text	Uses IR to get relevant text, uses IE to extract info
Information Retrieval (IR)	Find and rank relevant documents	Helps text mining by giving it the right docs
Information Extraction (IE)	Pull out specific details from text	A tool inside text mining

Text retrieval methods

When you ask a question or type a search (like "best smartphones"), the system needs to retrieve relevant documents. There are **two types of text retrieval methods**:

1. Document Selection Methods

Here, the system uses **Boolean logic** (AND, OR, NOT) to find documents that exactly match the user's conditions.

- **Example Queries**:
    - "mobile AND cheap"
    - "data science OR machine learning"
    - "coffee NOT tea"
- **Drawbacks**:
    - Requires exact logic
    - Not beginner-friendly
    - Not flexible (no ranking)
- **Use Case**: Works well if the user knows **exactly** what they want.

2. Document Ranking Methods

Instead of just selecting matching documents, this method:

- Finds **all possibly relevant** documents
- **Ranks** them by **how relevant** they are
- Like Google search ranks results — most relevant at the top.

- The system gives **scores** to documents based on how well they match your query.

8. Explain how web structure mining is different from web usage mining and web content mining? write a CLEVER algorithm for web structure mining.

1. Web Content Mining

- **What it is:** Extracting useful information from the **actual content** of web pages.
- **Focus:** Text, images, videos, metadata, etc.
- **Example:** Searching for "healthy recipes" on Google and getting actual articles, blog posts, or videos related to recipes.
- **Techniques Used:** Text mining, NLP (Natural Language Processing), multimedia mining.

2. Web Structure Mining

- **What it is:** Discovering relationships and structures from the **hyperlink structure** of the web.
- **Focus:** The **links between web pages**—like a web graph.
- **Example:** Figuring out which websites are **authoritative** (like Wikipedia or a government site) by looking at how many other pages link to it.
- **Techniques Used:** Graph theory, link analysis algorithms like **PageRank** or **CLEVER**.

3. Web Usage Mining

- **What it is:** Mining data from **web user behavior**—how users interact with websites.
- **Focus:** Clickstreams, browsing patterns, session logs.
- **Example:** Netflix analyzing your viewing habits to suggest movies.
- **Techniques Used:** Log file analysis, pattern recognition, clustering.

Aspect	Web Content Mining	Web Structure Mining	Web Usage Mining
Focus	Page content	Links between pages	User behavior
Input Data	Text, media, metadata	Hyperlinks	Web server logs, cookies, sessions

Aspect	Web Content Mining	Web Structure Mining	Web Usage Mining
Main Goal	Extract information	Identify authoritative/hub pages	Understand user preferences
Example Task	Extract product details	Find most important websites	Suggest products to a user

CLEVER Algorithm

The **CLEVER algorithm** is a famous algorithm used in **Web Structure Mining**. It helps find **authoritative pages** and **hub pages**.

What are Authorities and Hubs?

- **Authority Page**: A page that is a **credible source** of information (ex: official documentation, Wikipedia).
- **Hub Page**: A page that **links to many authoritative pages** (like a curated list of resources or a directory).
- A good hub points to good authorities, and a good authority is pointed to by good hubs.

Goal of CLEVER:

To **score** each page in a set of web pages with:

- **Authority Score**
- **Hub Score**
  These scores help us **rank** the pages.

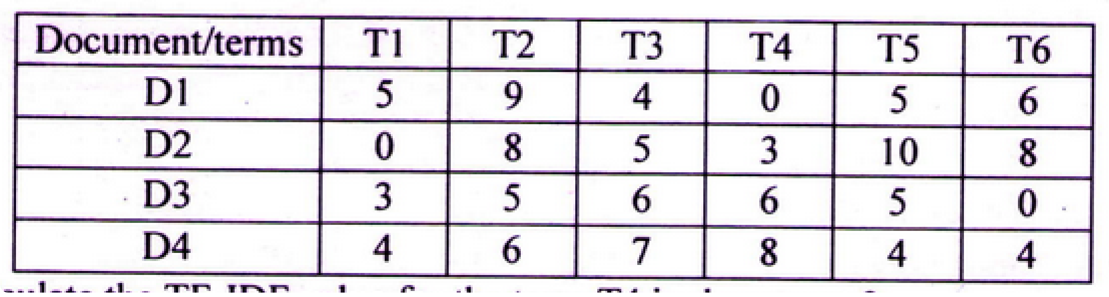
Basic Idea (How it Works):

1. **Start with a search query** (e.g., "Machine Learning").
2. Collect a set of relevant pages from a search engine — this is the **root set**.
3. Add pages that **link to** or are **linked from** the root set — this becomes the **base set**.
4. Treat all pages in this base set as **nodes in a graph**.
5. Use **iterative calculations** to assign **hub and authority scores** to each page.

Algorithm Steps

1. **Initialize:**
    1. Set hub and authority scores of all pages to 1.
2. **Authority Update Rule:**
    1. Each page's authority score = **sum of the hub scores** of all pages linking **to** it.
3. **Hub Update Rule:**
    1. Each page's hub score = **sum of the authority scores** of all pages it **links to**.
4. **Normalize the Scores** to prevent values from growing too large.
5. **Repeat** the update steps until the scores **converge** (i.e., stop changing significantly).

9.Term frequency matrix given in the table shows the frequency terms per document



Document/terms	T1	T2	T3	T4	T5	T6
D1	5	9	4	0	5	6
D2	0	8	5	3	10	8
D3	3	5	6	6	5	0
D4	4	6	7	8	4	4

Calculate the TF-IDF value for the term T4 in document 3.

Step 1: What is TF-IDF?

Think of **TF-IDF** like a score that tells us **how important a word (or term) is in a document**, compared to other documents.

- **TF (Term Frequency)** – How many times the word appears in this document?
- **IDF (Inverse Document Frequency)** – Is this word special? Or is it found in almost every document?

We combine both to find how meaningful the word is.

Step 2: Find TF (Term Frequency of T4 in D3)

- Look at the table. For **T4 in D3**, the number is:
- **TF = 6**
- That means the word T4 appears **6 times** in Document 3.

Step 3: Find how common T4 is in all documents (IDF part)

Now, we check **how many documents** mention T4:

- D1: ❌ T4 = 0 → Not used
- D2: ✅ T4 = 3 → Used
- D3: ✅ T4 = 6 → Used
- D4: ✅ T4 = 8 → Used
- So T4 is used in **3 out of 4 documents**.
- This means T4 is **not very rare**, but also **not used everywhere**.
- When a term appears in many documents, it's considered **less special**.

We use this to calculate a small number (called IDF) using a formula, but you can just think:

📌 **IDF for T4 = 0.1249**
(This was calculated earlier using a log formula — you don't need to memorize that now.)

Step 1: Term Frequency (TF)

- We need to find the TF first

Document	T4
D3	6

So,
TF(D3, T4) = 6

Step 2: Inverse Document Frequency (IDF)

We need:

- Total number of documents N=4N = 4N=4
- Number of documents in which T4 appears.

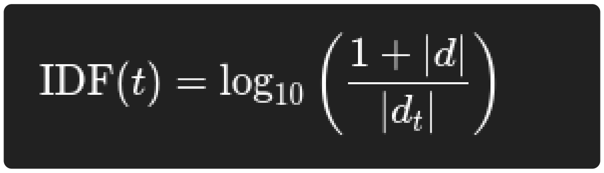
Document	T4
D1	0 → Not present
D2	3 → Present

Document	T4
D3	6 → Present
D4	8 → Present

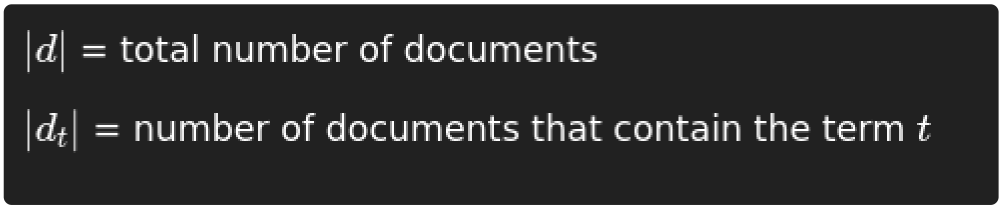
So,

df(T4) = 3

Now use the IDF formula:

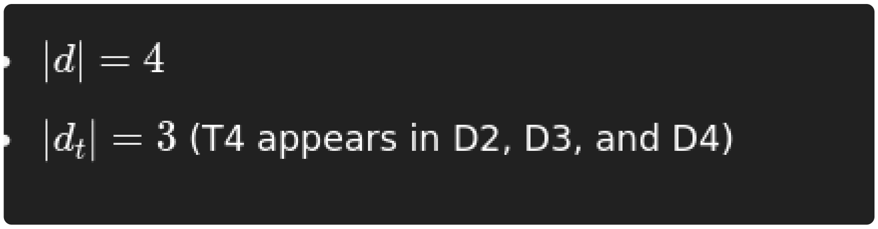


$$\mathrm{IDF}(t) = \log_{10}\left(\frac{1+|d|}{|d_t|}\right)$$

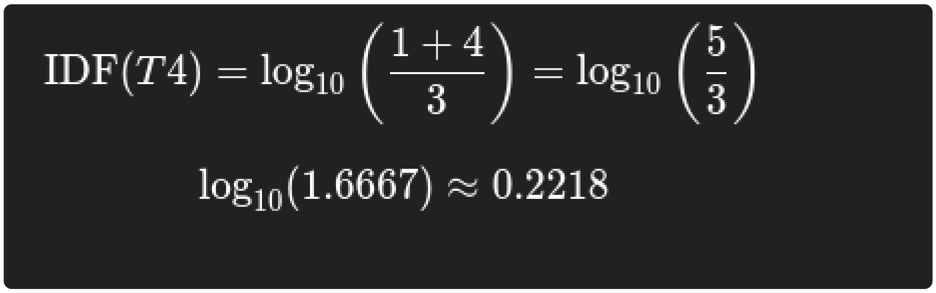


$|d|$ = total number of documents

$|d_t|$ = number of documents that contain the term t



- $|d| = 4$
- $|d_t| = 3$ (T4 appears in D2, D3, and D4)

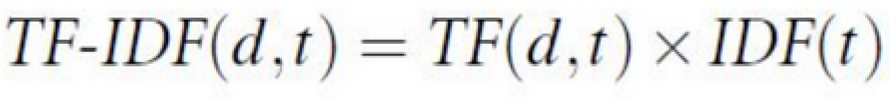


$$\mathrm{IDF}(T4) = \log_{10}\left(\frac{1+4}{3}\right) = \log_{10}\left(\frac{5}{3}\right)$$

$$\log_{10}(1.6667) \approx 0.2218$$

Step 3: Calculate TF-IDF

The formula for TF-IDF is



$$TF\text{-}IDF(d,t) = TF(d,t) \times IDF(t)$$

$$\text{TF-IDF}(D3, T4) = \text{TF} \times \text{IDF} = 6 \times 0.2218 = 1.3308$$

Step 4: Final answer

$$\boxed{\text{TF-IDF}(D3, T4) \approx 1.33}$$

10. List and explain the different data structures used for web usage mining?

What is Web Usage Mining?

Web Usage Mining is the process of **analyzing user behavior** by studying logs such as:

- Clickstreams
- Browser history
- Session data

To make sense of this data, we need efficient **data structures** to store and process **usage patterns**.

Data Structures Used in Web Usage Mining

To efficiently **track and discover patterns**, especially sequences of web pages visited by users, the following data structures are commonly used:

1. Trie (Prefix Tree)

What is a Trie?

- A **tree-based** data structure used to store **strings/sequences**.
- Each **path from root to leaf** represents a **complete sequence** (e.g., a user's page visit pattern).
- **Characters/Steps** are stored on **edges**.

- **Common prefixes are shared**, which makes them space-efficient (compared to naive string storage).

How It Helps in Web Usage Mining:

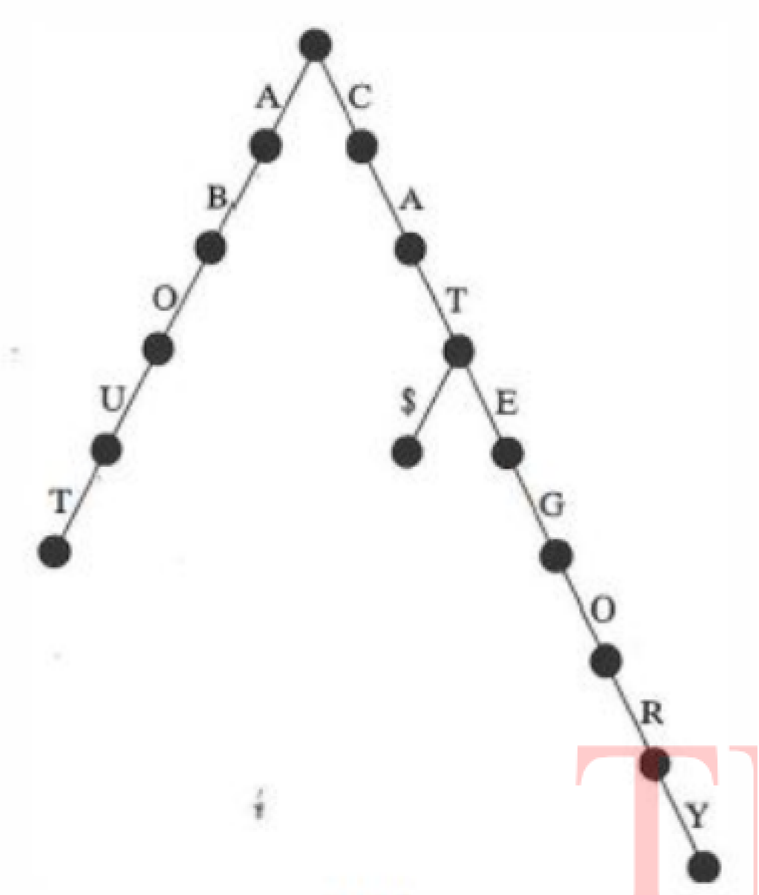
- It keeps track of **frequently occurring patterns** like:
  - "Homepage → Products → Checkout"
- Enables **fast pattern matching** and retrieval of sequences.
- Helps to **identify frequent paths** or clickstreams used by users.

Example:

For user navigation sequences:

- "ABOUT"
- "CAT"
- "CATEGORY"

  A standard trie would look like this:



(a) Trie

Problem with Standard Trie:

- If strings are **long and share many prefixes**, there will be many nodes with only one child.
- This **wastes memory**.

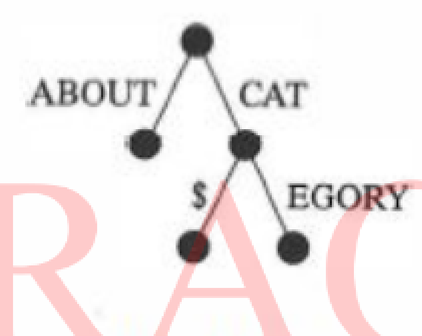
2. Compressed Trie / Suffix Tree

What is a Suffix Tree?

- A **compressed version** of a trie.
- Also known as **compact trie**.
- Combines **nodes with a single child** into a **single edge** labeled with a substring.

Why Use It?

- Saves memory.
- Still supports **fast lookups** and **pattern matching**.



(b) Suffix tree

11. Write any three applications of web usage mining and explain

1. Personalization

Personalization involves **adapting a website's content or layout** based on a specific user's behavior and preferences.

How Web Usage Mining helps:

- It tracks **previously visited pages** and **user navigation patterns**.
- Based on this, it can **predict what the user might want next**.
- Personalized suggestions can be shown like:

    - “You might also like...”
    - “Recommended for you…”

Real-life Example:

If a user frequently visits tech-related news articles, the homepage can be customized to show tech news at the top next time.

2. Improving Website Design

By analyzing how users interact with a website, designers can **identify problems** and **optimize user experience**.

How Web Usage Mining helps:

- Tracks which pages users **leave quickly (bounce)**.
- Finds pages where users **get stuck or drop off**.
- Helps identify which **navigation paths** are most/least used.

Example:

If most users never go beyond the product listing page, the website might need a better **call-to-action** or **more intuitive navigation** to the checkout page.

3. Business Intelligence and Marketing

Web usage data can be used to make **data-driven business decisions** for increasing sales and improving marketing strategies.

How Web Usage Mining helps:

- Identifies **user groups based on behavior** (e.g., buyers vs. browsers).
- Tracks which **ads, banners, or offers** users respond to most.
- Helps in **targeted advertising** and **product placement**.

Example:

An e-commerce site may discover that users who search for "budget laptops" often end up buying mid-range ones. The business can then **highlight mid-range laptops** more prominently.

12. Explain the different traversal patterns and discovery methods in web usage data.

Different traversal patterns

Traversal patterns describe **how users move through a website** — from one page to another.

1. Sequential Patterns

- Shows the **order** in which users access pages.
- Example: A → B → C (User visits Page A, then B, then C)
- Useful for **predicting next pages** a user might visit.

2. Frequent Patterns

- Identifies **commonly visited combinations** of pages.
- Not necessarily in order.
- Example: {A, B, C} is a frequent pattern if many users visit all three (in any order).
- Helps in **recommendation systems** and **caching strategies**.

3. Cyclic Patterns

- Detects if users **revisit certain pages** repeatedly.
- Example: A → B → A → C
- Indicates that users might be **confused** or that page A is a **hub** or **important page**.

4. Path Traversal Patterns

- Tracks the **complete navigation path** from entry to exit.
- Example: Home → Products → Cart → Exit
- Helps in analyzing **conversion funnels** and **drop-off points**.

Discovery Methods in Web Usage Mining

These are techniques used to **analyze and find patterns** from web usage data (usually from log files or session data).

1. Association Rule Mining

- Finds **relationships between pages**.
- Example: “Users who visited page A also visited page B with 80% probability.”
- Helps in **link recommendations** and **advertisement placement**.

2. Clustering

- Groups **similar user sessions** together.
- Based on browsing behavior (like time spent, pages visited).
- Useful in **user segmentation** and **personalization**.

3. Classification

- Assigns user sessions into **predefined categories**.
- Example: Classify users as **buyers** or **browsers**.
- Helpful for **targeted marketing**.

4. Sequential Pattern Mining

- Discovers **frequent access sequences**.
- Useful to **predict next user actions**.
- Example: If many users follow A → B → C, the system can suggest C after B.

13. Describe web content mining techniques.

1. Unstructured Text Mining

Used when content is **plain text** (without a defined structure), like articles, blogs, comments.

Techniques:

- **Information Extraction (IE):**
  - Extract structured information (like names, places, dates) from unstructured text.
- **Natural Language Processing (NLP):**
  - Used to understand the **meaning** and **context** of the text (e.g., sentiment analysis, summarization).
- **Text Classification:**

    - Assigns categories to content.
    - Example: Classify a blog as "tech", "health", or "news"
- **Text Clustering:**
    - Groups similar texts together (like news articles on the same topic).
- **Keyword/Concept Extraction:**
    - Finds important terms or topics from the text.

2. Structured Data Mining

Used when content is organized in **tables, lists, or records**.

Techniques:

- **Wrapper Induction:**
    - Used to extract data from structured parts of HTML (like product listings, tables).
- **DOM Tree Analysis:**
    - Analyzes the **Document Object Model (DOM)** structure to extract relevant parts (like prices, headings).
- **Pattern Matching (Regular Expressions):**
    - Used to find patterns in HTML source code (like emails, phone numbers, prices).

3. Semi-Structured Data Mining

Handles content that is **partially structured**, such as:

- XML files
- JSON from APIs
- HTML with tags but mixed text

Techniques:

- **Tree-Based Mining:**
    - Uses tree-like structures (like XML) for mining.
- **Schema Extraction:**

- Identifies patterns or templates used across pages (e.g., product layout on shopping websites).

14. Discuss various text mining approaches and techniques in detail.

Text Mining is the process of extracting **meaningful information** and **patterns** from **large volumes of unstructured or semi-structured text data**.

1. Information Retrieval (IR)

IR is about **finding relevant documents** based on a user's query

Two main IR methods:

Method	Description
Document Selection	Uses Boolean logic (AND, OR, NOT) to filter documents.
Document Ranking	Ranks documents based on **relevance score** (used in Google, etc.).

2. Text Preprocessing Techniques

Before mining, text must be cleaned and converted into a usable format.

Step	Purpose
Tokenization	Break text into words or tokens.
Stop Word Removal	Remove common but uninformative words like "the", "is", "of".
Stemming	Reduce words to their base/root form (e.g., "running" → "run").
Lemmatization	More advanced than stemming; returns actual dictionary word.

3. Term Frequency and Weighting (TF-IDF)

TF-IDF helps to **measure the importance** of a term in a document relative to a collection.

Formula:

- **TF(d,t)** = Frequency of term *t* in document *d*
- **IDF(t)** = log (Total number of docs / Number of docs containing *t*)
- **TF-IDF(d,t)** = TF × IDF
  TF-IDF increases with term frequency in a document and decreases with its frequency across all documents.

4.Text Classification

Assigns documents to predefined categories.

Examples:

- Spam vs. Non-Spam
- News topics like Politics, Sports, Health, etc.

5. Information Extraction (IE)

Extract structured information from unstructured text.
Tasks include:

- Named Entity Recognition (NER): Identify names, places, dates.
- Relationship Extraction: Discover links between entities (e.g., X works for Y).