Data, Information and Knowledge

Data: Often represented in simple structures like arrays, tables, or raw files. These structures are basic and hold unprocessed facts.

Information: Represented in more complex structures such as databases, spreadsheets, or data warehouses. These structures organize and provide context to data.
Knowledge: Typically represented in sophisticated structures like knowledge bases, ontologies, or expert systems. These structures are designed to capture and manage insights, rules, and relationships derived from information.
Relationship:

Data to Information: Data serves as the foundational layer from which information is derived through processing and organization.
Information to Knowledge: Information is analyzed and synthesized, often combined with experience and expertise, to create knowledge that provides actionable insights.
Control:

Control focuses on ensuring accurate collection and storage of raw data.
Information: Control involves the organization and presentation of information to ensure it is meaningful and accessible.
Knowledge: Control involves applying and contextualizing knowledge to guide decision-making and actions based on information and experience.
This updated table provides a comprehensive comparison of Data, Information, and Knowledge, including suitable data structures and their relationships and controls.

Parameter	Data	Information	Knowledge
Definition	Raw, unprocessed facts and figures without context.	Data that has been processed or organized to provide context.	Insight or understanding gained from processing information and experience.
Input	Raw observations or measurements (e.g., numbers, text).	Processed or structured data (e.g., tables, reports).	Processed information combined with experience and expertise (e.g., best practices).
Output	Unprocessed facts or figures.	Contextualized and meaningful content (e.g., reports, summaries)	Practical insights, conclusions, or actionable recommendations.
Process of Obtaining	Collecting raw facts or figures (e.g., sensors, surveys).	Organizing and analyzing data to make it meaningful (e.g., data aggregation, reporting).	Applying analysis, experience, and context to information to derive understanding (e.g., synthesis, learning).
Suitable Examples	- Temperature readings - Survey responses - Transaction records	Sales report - Weather forecast - Customer feedback analysis	Business strategy based on market trends - Expert opinions - Lessons learned from project management

Suitable Data Structure	- Arrays Tables - Raw files	Databases - Spreadsheets - Data warehouses	Knowledge bases - Ontologies - Expert systems
Relationship	Data is the raw material from which information is derived.	Information is the processed and contextualized form of data that can be used to generate knowledge.	Knowledge is derived from analyzing and interpreting information, often integrating multiple sources of information and experience.
Control	Control is typically limited to data collection and storage mechanisms.	Control involves the organization, access, and presentation of information to make it useful.	Control includes the application and contextualization of knowledge, often guiding decisions and actions based on information and experience.

Database VS DBMS

DBMS: Handles the input of data through various interfaces, such as forms, APIs, or data import tools. Produces organised and formatted data outputs, including reports and query results, with management tools for data access.Provides the functionality to query, retrieve, and manipulate data using languages like SQL or other query tools.Provides a comprehensive system for data DBMSt, including storage, retrieval, and manipulation. Examples include MySQL, PostgreSQL, and SQLite, which are open-source tools used to manage databases.The following table compares a database(a structured collection of data) with a DBMS(a software system used to manage and interact with that data).

Parameters	Database (DB)	Database Management System (DBMS)
Input	Raw data or records stored in a structured format (e.g., tables, files)	Structured data entry, manipulation, and management via various interfaces
Output	Stored data as-is, with no inherent processing or analysis	Organised, accessible data, and outputs such as reports or queries results
Query	Not applicable directly; data itself doesn't support querying without a DBMS	Supports queries (e.g., SQL) for data retrieval, updating, and management
Application Type	Data storage and retrieval in a structured form	Provides tools and functionality to manage, retrieve, and manipulate data
Open Source Examples	Not applicable as a database is a component, not a software tool	MySQL, PostgreSQL, SQLite

DBMS VS Data Mining

Input

DBMS: Focuses on structured data entered by users or applications. Provides structured and organised data or reports based on user queries.Uses SQL (Structured Query Language) to manage and query the data.Primarily used for data storage, organisation, and retrieval.Includes MySQL, PostgreSQL, and SQLite which are widely used for data management.

Data Mining: Can handle various types of data, including unstructured or semi-structured data.Outputs insights, patterns, and predictive models derived from analysing the data. Employs algorithms and models for pattern discovery rather than traditional querying. Focuses on analysing data to uncover patterns and make predictions.Includes tools like Weka, Orange, R (with packages for data mining), and KNIME, which are used for analysis and pattern discovery.

The following table presents a clear overview of how DBMS and Data Mining differ in terms of their functions and applications.

Parameter	**Database Management System (DBMS)**	**Data Mining**
Input	Structured data from various sources (e.g., tables, forms)	Data from various sources, including databases, spreadsheets, or text files
Output	Organised and manageable data (e.g., tables, views, reports)	Patterns, insights, or models (e.g., associations, clusters, predictions)
Query	SQL queries for data retrieval, updates, and management	Specialised queries or algorithms for discovering patterns (e.g., clustering, classification)
Application Type	Data storage, retrieval, and management	Data analysis, pattern recognition, and predictive analytics
Open Source Examples	MySQL, PostgreSQL, SQLite	Weka, Orange, R, KNIME

Database VS Knowledge Base

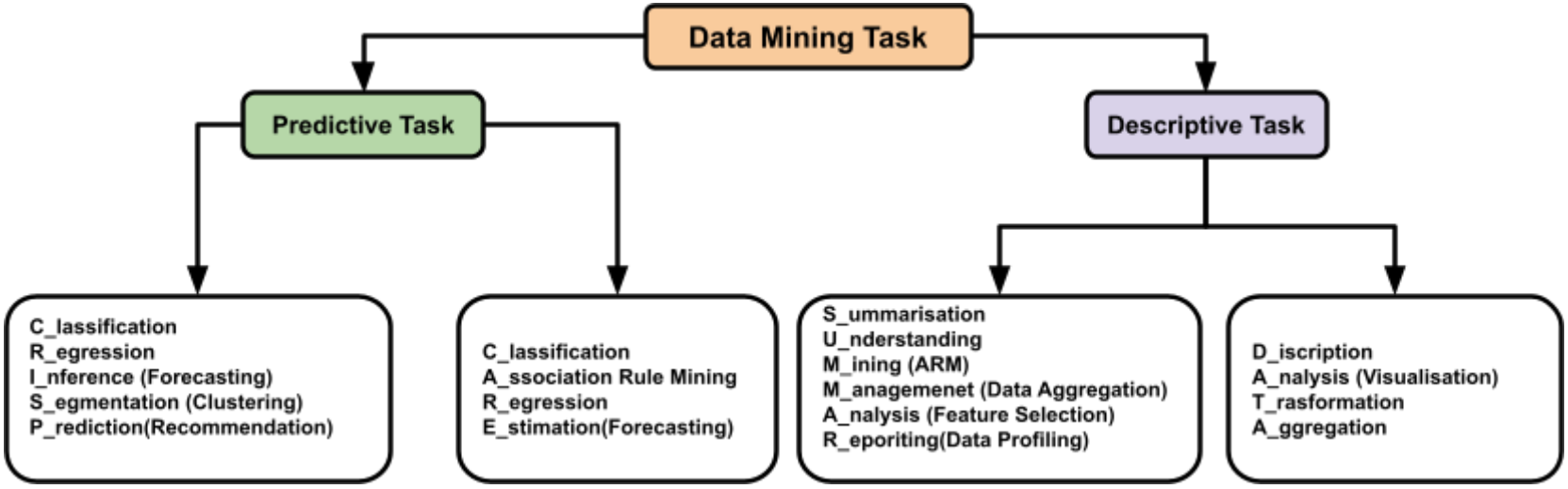
Database: A system for storing and managing structured data in tables, often used for transactional processes and data management.Typically receives structured data from various sources, such as user inputs, applications, or data imports. Outputs are often detailed and structured data reports or query results.Uses query languages like SQL to perform specific data retrieval and manipulation.Used for managing and organising data within various applications, including transactional systems and data warehouses. Examples include MySQL, PostgreSQL, and SQLite, which are used for data management.

Knowledge Base (KB): A system designed to store and manage knowledge, including both structured and unstructured information, to aid in decision-making and problem-solving.Includes a mix of structured data (e.g., FAQs) and unstructured content (e.g., expert insights, articles). Outputs include relevant information, solutions, and recommendations based on the queries or searches performed. Uses search functionalities to find relevant information based on keywords or topics. It Focuses on providing support and information for decision-making and problem resolution, often used in customer service and knowledge management.
Examples include MediaWiki (used for collaborative knowledge sharing), DokuWiki (simple wiki software), and OpenKB (knowledge base management system).
This table outlines the fundamental differences between databases and knowledge bases, focusing on their definitions, inputs, outputs, querying methods, applications, and open-source examples.

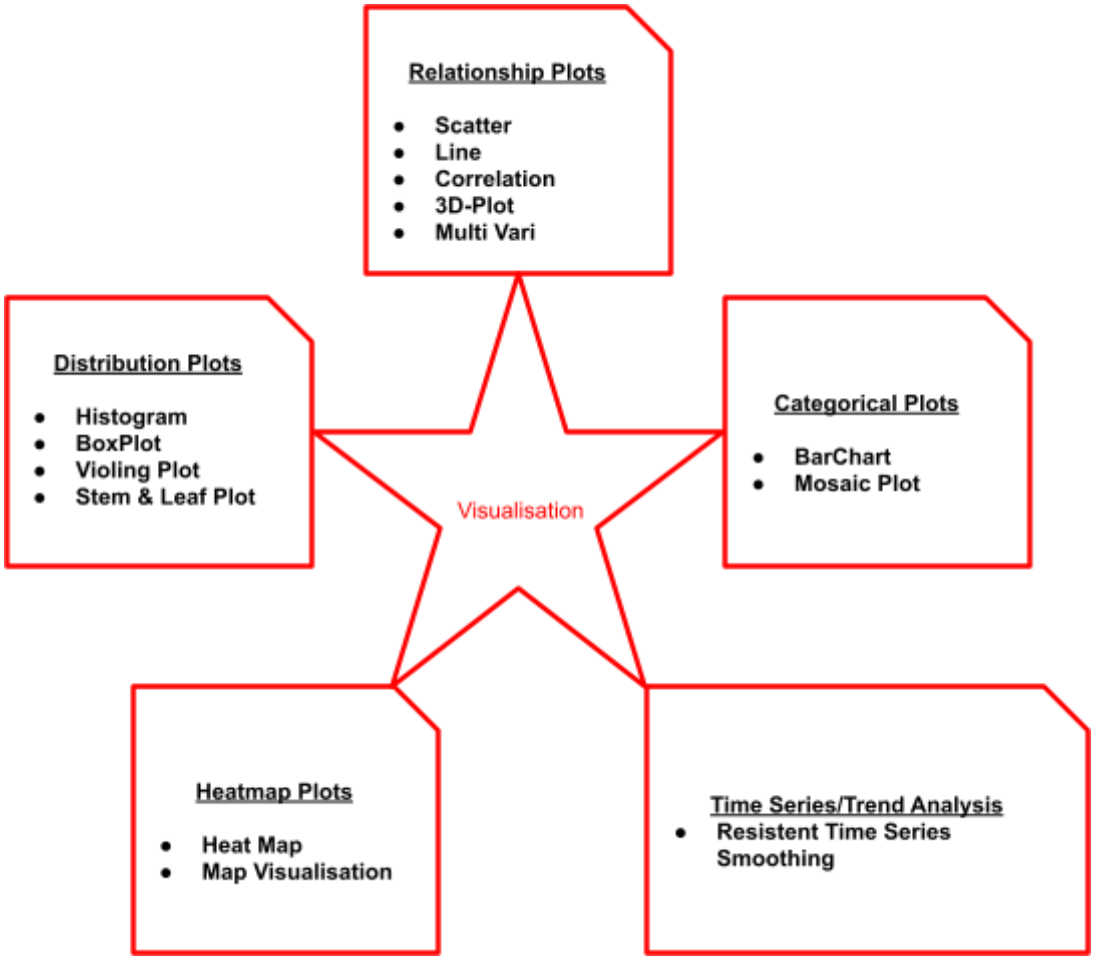
The following comparison table compares a Database (DB) with a Knowledge Base (KB) based on various parameters:

Parameter	Database (DB)	Knowledge Base (KB)
Definition	A structured collection of data stored and accessed electronically.	A repository of structured and unstructured information used to support decision-making, problem-solving, and learning.
Input	Raw data is entered into tables and records, typically structured and quantitative.	Information and knowledge, which can include structured data, documents, FAQs, and expert insights.

Output	Structured data outputs, such as reports and data queries.	Useful insights, solutions, and recommendations based on the stored information.
Query	Supports querying using languages like SQL to retrieve and manipulate data.	Queries or searches are based on keywords or topics to retrieve relevant information or answers.
Application Type	Data storage, retrieval, and management for various applications.	Knowledge sharing, decision support, and problem-solving.
Open Source Examples	MySQL, PostgreSQL, SQLite	MediaWiki, DokuWiki, OpenKB


Data Mining Task
Predictive Task
Descriptive Task
C_lassification
R_egression
I_nference (Forecasting)
S_egmentation (Clustering)
P_rediction(Recommendation)
C_lassification
A_ssociation Rule Mining
R_egression
E_stimation(Forecasting)
S_ummarisation
U_nderstanding
M_ining (ARM)
M_anagemenet (Data Aggregation)
A_nalysis (Feature Selection)
R_eporiting(Data Profiling)
D_iscription
A_nalysis (Visualisation)
T_rasformation
A_ggregation

Some types of Plots



Plot Type	Definition	Characteristics	Application	Python Syntax	Example
Scatter Plot	A plot that displays points based on two variables.	- Shows relationships between variables. - Points scattered in a 2D space.	- Correlation analysis - Outlier detection	plt.scatter(x, y) sns.scatterplot(x, y)	plt.scatter(x, y) sns.scatterplot(data=df, x='var1', y='var2')
Line Plot	A plot that shows data points connected by lines.	- Displays trends over time. - X-axis usually represents time.	- Time series analysis - Trend identification	plt.plot(x, y) sns.lineplot(x, y)	plt.plot(x, y) sns.lineplot(data=df, x='time', y='value')
Bar Chart	A plot that presents categorical data with rectangular bars.	- Bars can be vertical or horizontal. - Height/length represents value.	- Comparing categorical data - Distribution analysis		plt.bar(x, height) sns.barplot(data=df, x='category', y='value')
Box Plot	A plot that displays the distribution of data based on quartiles.	- Shows median, quartiles, and potential outliers.	- Distribution analysis - Identifying outliers	plt.boxplot(data) sns.boxplot(x, y)	plt.boxplot(data) sns.boxplot(data=df, x='category', y='value')
Correlogram	A plot that visualises the	- Matrix format. - Uses color intensity to	- Understanding relationships	sns.heatmap(corr_matrix)	sns.heatmap(corr_matrix, annot=True)

	correlation matrix of variables.	represent correlation strength.	- Multivariate data analysis		
Heatmap	A plot that represents data with varying color intensities.	- Uses a color gradient to represent data magnitude.	- Visualizing matrix data - Comparing data intensity across two dimensions	sns.heatmap(data)	sns.heatmap(data, cmap='viridis', annot=True)
Histogram	A plot that shows the distribution of a dataset.	- Bars represent frequency of data intervals. - Useful for continuous data.	- Distribution analysis - Frequency analysis	plt.hist(data) sns.histplot(data)	plt.hist(data, bins=30) sns.histplot(data=df['value'], kde=True)
Map Visualization	A plot that displays data on geographical maps.	- Integrates data with geographic locations. - Choropleth maps or point-based maps.	- Spatial data analysis - Geographic distributions	plt.scatter(lon, lat) folium.Map(location=[lat, lon])	plt.scatter(lon, lat) folium.Map(location=[45.5236, -122.6750], zoom_start=13)
Mosaic Plot	A plot that shows the relationship between categorical variables.	- Proportional rectangles represent the size of categories.	- Visualizing contingency tables - Categorical data comparison	from statsmodels.graphics.mosaicplot import mosaic	mosaic(df, ['Category1', 'Category2'])

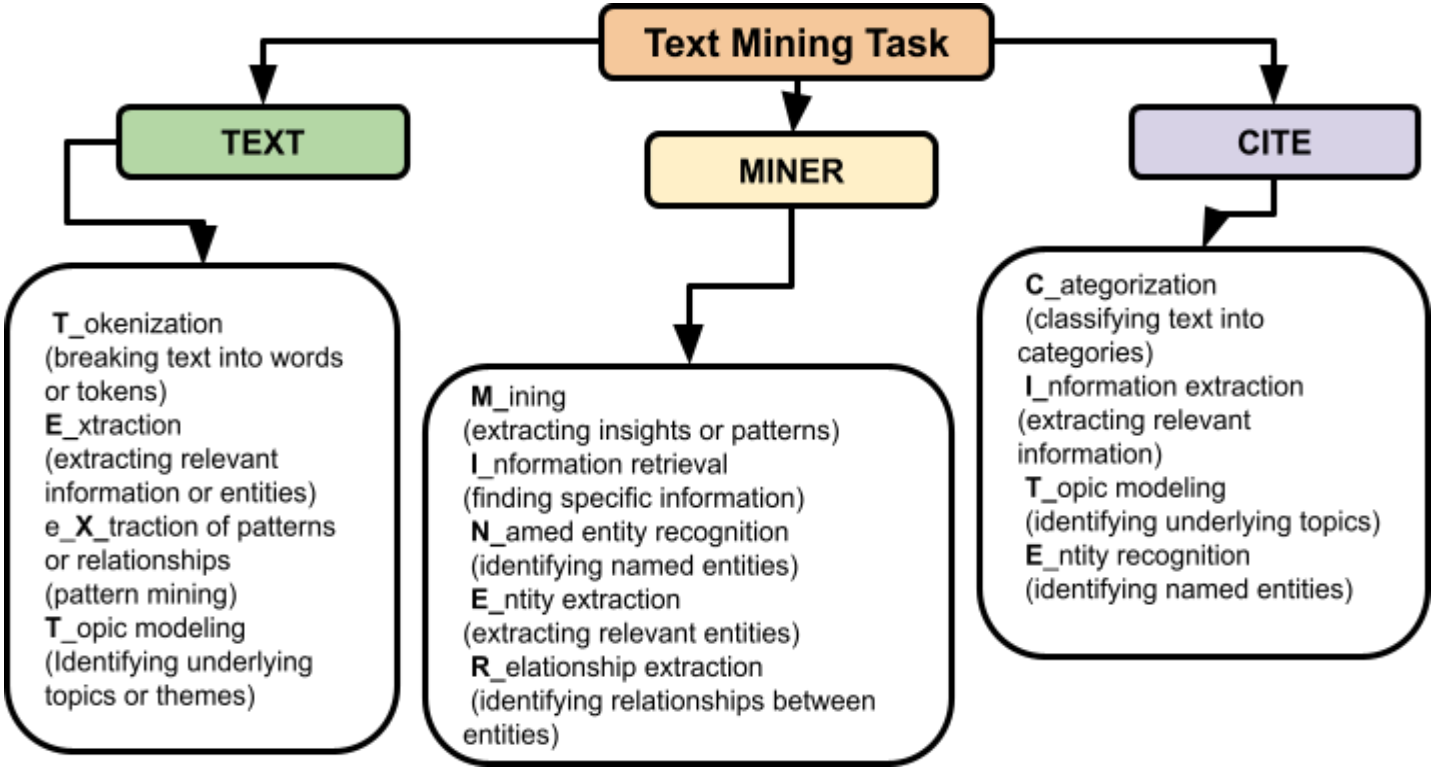
3D Plot	A plot that displays data in a three-dimensional space.	- Adds depth to traditional plots. - Can be scatter, surface, or wireframe plots.	- Visualising 3D data - Surface analysis	ax = plt.axes(projection='3d') ax.plot3D(x, y, z)	ax = plt.axes(projection='3d') ax.scatter3D(x, y, z, c=z, cmap='Greens')
Stem and Leaf Plot	A plot that displays data distribution while retaining original values.	- Combines aspects of a histogram and box plot. - Shows individual data points.	- Quick overview of distribution - Small datasets	Not directly available in Python; can be manually created using text-based methods	Manual creation using loops and print statements
Violin Plot	A plot that shows the distribution and density of data.	- Combines features of a box plot and a density plot. - Symmetric or asymmetric distribution.	- Distribution comparison - Visualisation of data density	sns.violinplot(x, y)	sns.violinplot(data=df, x='category', y='value')
Multi-Vari Chart	A plot that displays the variation within groups over time.	- Visualises the variability of multiple factors. - Can include means, medians, or ranges.	- Analysis of multiple variables - Quality control	Not directly available in Python; can be simulated using combinations of	Combine plt.plot() and plt.boxplot() for simulation

				box plots and line plots	
Rootogram	A plot that compares observed and expected frequencies.	- Visualises goodness of fit for a statistical model. - Bars extend above/below a baseline.	- Model fitting - Comparing observed vs. expected distributions	from statsmodels.graphics.gofplots import ProbPlot	pp = ProbPlot(data, dist="norm") pp.rootogram()
Resistant Time Series Smoothing	A technique for smoothing time series data to reduce noise.	- Robust to outliers. - Emphasises the underlying trend.	- Time series analysis - Reducing noise in volatile datasets	from statsmodels.robust_scale import hampel	smoothed_data = hampel(data, window_size=7, n_sigmas=3)

Notes:

- **Manual Implementations**: Some of the advanced or less common plots like the **Stem and Leaf Plot** and **Multi-Vari Chart** do not have direct implementations in standard Python plotting libraries. They can be manually created or simulated using a combination of existing functions.
- **Statistical Tools**: **Rootogram** and **Resistant Time Series Smoothing** are more specialised tools and may require specific statistical libraries such as statsmodels.
- **Python Libraries**: The examples often use matplotlib, seaborn, and statsmodels. Other libraries may be used for more specialised or interactive visualisations.

Text Mining



TEXT:

- T: Tokenization (breaking text into words or tokens)
- E: Extraction (extracting relevant information or entities)
- X: eXtraction of patterns or relationships (pattern mining)

- T: Topic modeling (identifying underlying topics or themes)

MINER:
- M: Mining (extracting insights or patterns)
- I: Information retrieval (finding specific information)
- N: Named entity recognition (identifying named entities)
- E: Entity extraction (extracting relevant entities)
- R: Relationship extraction (identifying relationships between entities)

CITE:
- C: Categorization (classifying text into categories)
- I: Information extraction (extracting relevant information)
- T: Topic modeling (identifying underlying topics)
- E: Entity recognition (identifying named entities)

Sample Case Study: (Text Mining Task)
Goal: To classify written complaints or FIRs (First Information Reports) into applicable BNS (Broad National Categories) sections or IPC (Indian Penal Code) sections.
Consider the following complaint (Written Complaint)

"Theft of my mobile phone by an unknown person at MG Road on 2023-02-20. The phone was stolen when I was shopping at a store. The thief was wearing a black jacket and had a scar on his face."

Question: How text mining techniques, you can automate the process of classifying written complaints or FIRs into applicable BNS sections or IPC sections, making it easier to analyze and track crime patterns?
Answer: By using text mining techniques, you can automate the process of classifying written complaints or FIRs into applicable BNS sections or IPC sections, making it easier to analyze and track crime patterns.

Phase	Task	Illustration
Tokenization	Break down the text into individual words or phrases	- "Theft of mobile phone" - "Physical assault by unknown person" - "Cheating and fraud by business partner"
Extraction	Identify relevant information and entities	- "Theft" (entity: crime) - "Mobile phone" (entity: stolen item) - "Physical assault" (entity: crime) - "Unknown person" (entity: perpetrator) - "Cheating and fraud" (entity: crime) - "Business partner" (entity: perpetrator)
Topic Modeling	Identify underlying topics or themes	- "Property crime" - "Violent crime" - "White-collar crime"
Entity Recognition	Identify named entities	- "IPC 379" (entity: IPC section for theft) - "BNS 4.1" (entity: BNS section for property crime)
Relationship Extraction	Identify relationships between entities	- "Theft of mobile phone" is related to "IPC 379" - "Physical assault" is related to "BNS 4.2" (violent crime)
Classification	Classify the text into applicable BNS sections or IPC sections	- Complaint 1: BNS 4.1 (property crime) and IPC 379 (theft) - Complaint 2: BNS 4.2 (violent crime) and IPC 323 (physical assault) - Complaint 3: BNS 4.3 (white-collar crime) and IPC 420 (cheating and fraud)

Proposed algorithms

Consider the following complaint (Written Complaint)

"Theft of my mobile phone by unknown person at MG Road on 2023-02-20. The phone was stolen when I was shopping at a store. The thief was wearing a black jacket and had a scar on his face."

In following table, various steps are shown that identifies the written complaint as related to IPC section 379 (theft) and BNS section 4.1 (property crime).

Phases of Text Mining	**Sample Outcome**
Text Preprocessing	- Tokenization: ["Theft", "of", "my", "mobile", "phone", "by", "unknown", "person", "at", "MG", "Road", "on", "2023-02-20", ...] - Stop word removal: ["Theft", "mobile", "phone", "unknown", "person", "MG", "Road", "2023-02-20", ...] - Stemming/Lemmatization: ["Theft", "mobile", "phone", "unknown", "person", "MG", "Road", "2023-02-20", ...]
Named Entity Recognition (NER)	- Entities: ["Theft" (crime), "mobile phone" (stolen item), "unknown person" (perpetrator), "MG Road" (location), "2023-02-20" (date)]
Part-of-Speech (POS) Tagging	- POS tags: ["Theft" (noun), "mobile" (adjective), "phone" (noun), "unknown" (adjective), "person" (noun), ...]
Dependency Parsing	- Relationships: ["Theft" -> "of" -> "mobile phone", "mobile phone" -> "was stolen", ...]
IPC/BNS Section Identification	- Knowledge graph: [IPC 379 (theft), BNS 4.1 (property crime), ...] - Matching: ["Theft" -> IPC 379, "mobile phone" -> BNS 4.1, ...]
Ranking and Filtering	- Ranked IPC/BNS sections: [IPC 379 (high confidence), BNS 4.1 (high confidence), ...]
Post-processing	- Validated IPC/BNS sections: [IPC 379 (theft), BNS 4.1 (property crime)]

Suitable Data Structures

Following are some applicable data structures for the algorithm:

Input /Data	**Organization of data and Possible Operations**
Text Data	- String arrays or lists for tokenized text - Hash tables or dictionaries for word frequencies
Knowledge Graph	- Graph databases (e.g., Neo4j) for storing IPC/BNS sections and relationships - Hash tables or dictionaries for mapping keywords to IPC/BNS sections
Named Entity Recognition (NER)	- Lists or arrays for storing extracted entities - Hash tables or dictionaries for entity frequencies
Part-of-Speech (POS) Tagging	- Lists or arrays for storing POS tags - Hash tables or dictionaries for POS tag frequencies
Dependency Parsing	- Graph data structures (e.g., adjacency lists) for storing sentence dependencies - Hash tables or dictionaries for storing dependency relationships
IPC/BNS Section Identification	- Hash tables or dictionaries for mapping keywords to IPC/BNS sections - Lists or arrays for storing identified IPC/BNS sections
Ranking and Filtering	- Priority queues or heaps for ranking IPC/BNS sections - Hash tables or dictionaries for filtering out low-confidence sections
Additional data structures (Depending on the specific implementation and requirements.)	- Trie or prefix tree for efficient text matching - Suffix tree for efficient substring matching - Graph neural networks for learning relationships between IPC/BNS sections

Performance Metrics of Text Mining

अद्भुत लोग उत्कृष्ट रोबोट बनाकर शानदार उत्तर देते हैं, ताज़ी कुकीज़ खाने वाले अच्छे कीड़ों को रोकते हैं
"Awesome People Render Fantastic Answers Making Perfect Robots Keep Nice Bugs Eating Fresh Cookies"

A - ACC
P - PRE
R - REC
F - F1
A - AUC
M - MAP
P - PER
R - R@K
K - (no direct match, but can be associated with P@K and R@K)
N - NDCG
B - BLEU
E - (no direct match, but can be associated with ROUGE)
F - FLU
C - COH

1. ACC (Accuracy)
2. PRE (Precision)
3. REC (Recall)
4. F1 (F1-score)
5. ROC-AUC (no acronym, but can be referred to as AUC)
6. MAP (Mean Average Precision)
7. MRR (Mean Reciprocal Rank)
8. P@K (Precision at K)
9. R@K (Recall at K)
10. NDCG (Normalized Discounted Cumulative Gain)
11. BLEU (Bilingual Evaluation Understudy)
12. ROUGE (Recall-Oriented Understudy for Gisting Evaluation)
13. PER (Perplexity)

14. COH (Coherence)
15. FLU (Fluency)

1. Accuracy: Proportion of correctly classified or extracted instances.

2. Precision: Proportion of true positives (correctly classified or extracted instances) among all positive predictions.

3. Recall: Proportion of true positives among all actual instances.

4. F1-score: Harmonic mean of precision and recall.

5. ROC-AUC (Receiver Operating Characteristic - Area Under the Curve): Measures the model's ability to distinguish between classes.

6. Mean Average Precision (MAP): Evaluates ranking quality for information retrieval tasks.

7. Mean Reciprocal Rank (MRR): Measures the ability to retrieve relevant items.

8. Precision at k (P@k): Precision within the top-k retrieved items.

9. Recall at k (R@k): Recall within the top-k retrieved items.

10. Normalized Discounted Cumulative Gain (NDCG): Measures ranking quality, considering relevance and position.

11. BLEU score (Bilingual Evaluation Understudy): Evaluates text generation quality.

12. ROUGE score (Recall-Oriented Understudy for Gisting Evaluation): Measures text summarization quality.

13. Perplexity: Measures language model quality, lower values indicate better performance.

14. Coherence: Measures the logical consistency of extracted information.

15. Fluency: Measures the readability and grammatical correctness of generated text.

Annexure

Learning Outcomes

Some questions for data science understanding:

Basic

1. What is the definition of data?
2. What are the three types of data structures?
3. What is the purpose of a database management system?
4. What is the difference between a database and a knowledge base?
5. What is text mining?
6. What is the goal of topic modelling?
7. What is entity recognition?
8. What is relationship extraction?
9. What is the purpose of classification in text mining?
10. What is the difference between IPC and BNS sections?

Understanding

1. Can you explain the relationship between data, information, and knowledge?
2. How does a database management system organize data?
3. What is the difference between a scatter plot and a line plot?
4. How does text preprocessing work in text mining?
5. What is the purpose of named entity recognition?
6. Can you describe the difference between a knowledge base and a database?
7. How does topic modeling work?
8. What is the purpose of dependency parsing?

9. Can you explain the difference between IPC and BNS sections?
10. How does classification work in text mining?

Applying

1. How would you design a database for a specific application?
2. Can you create a scatter plot to represent a given dataset?
3. How would you preprocess text data for analysis?
4. Can you identify the entities in a given text?
5. How would you extract relationships between entities in a text?
6. Can you classify a given text into IPC or BNS sections?
7. How would you design a knowledge base for a specific domain?
8. Can you create a heatmap to represent a given dataset?
9. How would you apply topic modeling to a given text dataset?
10. Can you design a system for text mining?

Analyzing

1. Can you analyze the differences between various data structures?
2. How does the choice of database management system affect data organization?
3. Can you compare and contrast different types of plots?
4. What are the limitations of text preprocessing?
5. Can you evaluate the effectiveness of named entity recognition?
6. How does the design of a knowledge base impact its usability?
7. Can you analyze the results of topic modeling?
8. What are the challenges of dependency parsing?
9. Can you evaluate the accuracy of classification in text mining?
10. How does the choice of classification algorithm impact results?

Evaluating

1. Can you justify the choice of a specific database management system?
2. How does the selection of plots impact data interpretation?
3. Can you argue for or against the use of text preprocessing?
4. Can you evaluate the effectiveness of entity recognition?
5. How does the design of a knowledge base impact its effectiveness?
6. Can you justify the choice of topic modeling algorithm?
7. Can you evaluate the impact of dependency parsing on text analysis?
8. How does the choice of classification algorithm impact results?
9. Can you argue for or against the use of text mining?
10. Can you evaluate the overall effectiveness of a text mining system?

Creating

1. Can you design a new data structure for a specific application?
2. How would you create a new type of plot to represent data?
3. Can you develop a new algorithm for text preprocessing?
4. Can you design a system for entity recognition?
5. How would you create a knowledge base for a new domain?
6. Can you develop a new topic modeling algorithm?
7. Can you design a system for dependency parsing?
8. How would you create a new classification algorithm?
9. Can you design a new system for text mining?
10. Can you develop a new application for text mining?