Data Science
Qp format

- Unit 1 3 out of 4
- ~~1. Data types nominal ordinal / temporal or special data or compare data / info~~
- ~~2. Ide's feature of ide used for prog in python thoda case study types write about adv and disadvantage of ide and eda(exprolatory data analysis - handle missing values and data in ppt has given an example)~~
- ~~3. Decimal ka spacing two techniques or short note~~
- ~~4. Data mining and data warehouse application in respective fields or data science same kind of answer .~~
- ~~5. In data warehouse compare data warehouse and DBMS , architecture of data ware house adv and disadvantage create short note~~
- ~~6. Prac 1 basics of numpy and pandas so simple progs might come or he will give a SS of code spot errors and justify~~
- Eg how np. Eye and np. I is diff
    - Both np.eye and np.identity are used for creating identity matrices.
    - np.eye offers more flexibility with the k parameter to create diagonals with 1s at different positions.
    - np.identity is simpler and the preferred choice for the basic square identity matrix.
- Unit 2 .
- ~~1. Python code / libraries (numpy pandas) comparison how diff and give example~~
- ~~If question is demonstrate numpy and pandas can write two codes or one code with both in them~~
- ~~2. Security of data h remote data (aryan ka ppt in data science) comparison of authorisation and authentication if data~~
- ~~Precaution with data collection~~
- ~~Diff mechanism two way auth or OTP or password open ended can write anything~~
- ~~Can give small case study if taking data from us uk what precautions and privacy issues how to deal create short note~~
- ~~Prac 2 how to read exel file and csv and operation here data will not be given can assume write reading from data. CSV only commands file banana nahi haii~~
- ~~Prac 1 and 2 is part of theory prep properly~~
- ~~3. Data analysis diagram / phases tisha ppt~~
- ~~4. Graphs diff graphs imp of bar chart line chart histogram scattered plot and uska basic code just make two arrays x and y and use matplot~~
- ~~5.graphs why important give example advantages and show data in practical way open ended(feka maaro)~~
- Unit 3
- ~~Framework of data science all layers~~
- ~~Sabka short note advantage and disadvantage diagram~~
- ~~Buffer drum and role problem uska 6 phases and diagram~~
- ~~Fictional and non functional requirements~~
- ~~Refer kush waala ppt~~
- ~~Audit balance control layer~~
- ~~Draw diagrams wherever possible~~
- ~~Aws - ananya ka ppt but simple question importantance of AWS data set why more important why ppl go for AWS compared to other / personal collection of data~~

- ~~Diff data sets , gnom , landset , multimedia~~
- ~~3. Nosql - features of nosql include mongodb features~~
- ~~Simple example to create schema code and get and crud operations ka code~~
- ~~4. Ontology advantages disadvantages uska languages - owl , rdfs - schema basics of ontology~~
- Unit 4
- ~~1 fix on ID3 compulsory sum~~
- ~~1 on linear regression sum~~
- ~~1 Confusion Matrix sum formula / shortnote bhi aa skta hai give example and explain~~
- ~~ID3 ka steps bhi aa skta hai if not sum~~
- ~~Type 1 and type 2 errors~~
- ~~Examples~~
- ~~Short note on types of regression polynomial/ logical adv disadvantage and application areas and~~
- ~~simple python code data in list x and y just 5 - 6 line ka code in pracs~~
- ~~Can also have comparison in regression types type 1 type 2 mien case study aa skta~~
- ~~Supervised and unsupervised bias variance , under and overfitting~~
- ~~Binary classification, lazy learning, eager learning. 2 question are sums pkaa~~

Unit 1

Data
In data science, data refers to raw, unprocessed information that can be collected, analyzed, and used to extract knowledge, identify patterns, and make predictions. It's the foundation for all data science tasks. Data can come in various forms, each with its own characteristics and processing requirements.

Types of Data
Data can be broadly categorized into two main types: structured and unstructured.

Structured Data
Definition: Structured data is highly organized and follows a predefined format. It typically resides in relational databases or spreadsheets where each piece of data has a specific meaning and location. It's easy for computers to process and analyze due to its well-defined structure.
Examples:
- Customer information in a database (customer ID, name, address, purchase history)
- Financial data in spreadsheets (stock prices, transaction details)
- Sensor data with timestamps and values

Unstructured Data
Definition: Unstructured data is information that lacks a consistent format or organisation. It often contains text, images, audio, video, emails, social media posts, and other complex data types. While valuable for insights, unstructured data requires additional processing and techniques for analysis.
Examples:
- Text documents, emails, social media posts
- Images, videos, audio files

- Website content, sensor logs, network traffic data

Data Properties Comparison Table

Property	Structured Data	Semi-structured Data	Unstructured Data
Technology	Relational database tables	XML, RDF (Resource Description Framework)	Character & binary data
Transaction Management	Matured, various concurrency techniques	Adapted from DBMS, less mature	No transaction management or concurrency
Version Management	Versioning over tuples, rows, tables	Versioning over tuples or graphs (possible)	Versioned as a whole
Flexibility	Schema-dependent, less flexible	More flexible than structured, less than unstructured	Most flexible, no schema
Scalability	Difficult to scale database schema	Simpler scaling than structured	Most scalable
Robustness	Very robust	Newer technology, less widespread	-

Query Performance	Complex joins with structured queries	Queries over anonymous nodes possible	Only textual queries possible

Data vs information

Data	Information
Variables for developing ideas/conclusions	Meaningful data
Text and numerical values	Refined form of actual data
Independent (doesn't rely on information)	Relies on data
Measured in bits and bytes	Measured in meaningful units (time, quantity)
Can be structured (tabular, graph, data tree)	Can be structured (language, ideas, thoughts)
No inherent meaning	Meaningful context
Low-level knowledge	Second level of knowledge

Indirectly helps in decision making	Directly helps in decision making
Collection of facts with no meaning	Puts facts into context
Example: Student test score	Example: Average score of class

Qualitative vs Quantitative data

Qualitative vs. Quantitative Data

Feature	Qualitative Data	Quantitative Data
Definition	Descriptive data; focuses on qualities or characteristics	Numerical data; focuses on quantities or measurements
Collection Methods	Interviews, Focus Groups, Observations, Open-ended Surveys	Surveys, Experiments, Sensors, Existing Records
Data Format	Textual; narratives, quotes, images, audio/video recordings	Numerical; numbers, statistics, ratings on scales
Analysis Methods	Thematic analysis, coding, identifying patterns	Statistical analysis, calculations, creating charts and graphs

Purpose	Understand experiences, motivations, opinions	Measure, quantify, compare, predict
Examples	Customer feedback in text, interview transcripts	Sales figures, website traffic data, survey responses with numerical scales

Nominal vs. Ordinal Data

Feature	Nominal Data	Ordinal Data
Definition	Categorical data with no inherent order or ranking	Categorical data with a specific order or ranking
Examples	Hair color (blonde, brown, black), Blood type (A, B, AB, O), Country of origin	Customer satisfaction (very satisfied, satisfied, neutral, dissatisfied, very dissatisfied), Movie ratings (1 star, 2 stars, 3 stars, 4 stars, 5 stars)
Mathematical Operations	Limited; cannot perform calculations like addition, subtraction, or averaging	Possible; can determine "greater than," "less than," or "equal to," but not the difference between categories
Level of Measurement	Nominal (lowest level)	Ordinal (second level)

Analysis Methods	Frequency tables, Chi-square tests	Frequency tables, Chi-square tests, rankings, order statistics (median)
Insights	Identify groups, compare distributions	Identify groups, compare distributions, understand order or hierarchy

Discrete vs. Continuous Data

Feature	Discrete Data	Continuous Data
Definition	Data that can only take on whole, distinct values	Data that can take on any value within a specific range
Visualisation	Represented well with bar graphs or histograms (with bars)	Represented well with line graphs or histograms (without bars)
Examples	Number of people in a room (1, 2, 3, ...), Shoe size (5, 6, 7, ...), Exam scores (assuming whole numbers)	Temperature, Weight, Time, Distance
Measurement	Counted	Measured

Values Between Categories	Gaps or jumps exist between possible values	No gaps or jumps between possible values (theoretically infinite possibilities within the range)
Analysis Methods	Frequency tables, Chi-square tests, mode (most frequent value)	Statistical analysis (mean, median, standard deviation), regression analysis
Real-world Examples	Inventory counts, Number of website visitors per day, Number of correct answers on a multiple-choice test	Height, Weight, Speed, Temperature at a specific time

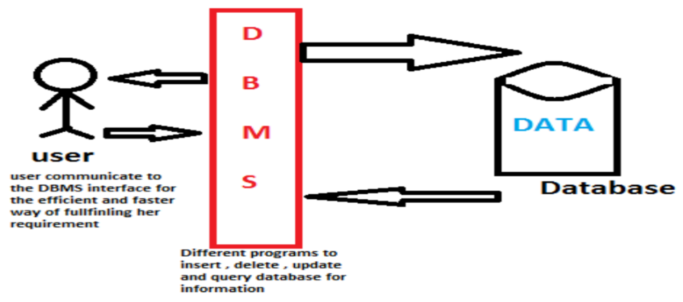
Further Classification of data

1. Demographic Data:
    - Description: This data describes the characteristics of a population or individual, like age, gender, income, education level, family size, occupation, etc.
    - Source: Collected through surveys, census data, customer profiles, loyalty programs.
    - Use Cases: Segmenting customers, targeted marketing campaigns, understanding customer needs and preferences.
2. Transactional Data:
    - Description: This data captures details about customer transactions, including products purchased, date and time of purchase, amount spent, payment method, etc.
    - Source: Point-of-sale systems, e-commerce platforms, financial records.
    - Use Cases: Analysing purchase patterns, identifying trends, recommending products, optimising pricing and promotions.
3. Web Behavior Data:
    - Description: This data tracks how customers interact with a website or app, including pages visited, time spent on each page, products viewed, search queries, etc.
    - Source: Web analytics tools (e.g., Google Analytics), clickstream data, user session recordings.
    - Use Cases: Personalising user experience, website optimization, understanding customer journeys, identifying popular products.
4. Data from Customer-Created Texts:
    - Description: This data encompasses text reviews, feedback, comments, social media posts, and other forms of written communication from customers.
    - Source: Online reviews, social media platforms, customer service interactions, chat logs.

- Use Cases: Sentiment analysis (understanding customer opinions), improving product/service offerings, identifying customer pain points, building brand loyalty.

DBMS

DBMS… way of data extraction


D
B
M
S
user
user communicate to the DBMS interface for the efficient and faster way of fullfinling her requirement
DATA
Database
Different programs to insert , delete , update and query database for information

Problems faced by current DBMS

- large quantities of data is generated /processed.
- data may get doubled in every say 3 months.
- Seeking knowledge from this massive data is most required.
- Fast developing in computer science and engineering techniques generates new demands.   To fulfill those demands we require to analyze the data
- Data Rich , Information Poor.. Raw data by itself does not provide much information.
- In today's life we require only significant data from which we can judge the customer's likings and strategies

Data Mining

Data mining is the multifaceted process of extracting valuable knowledge, patterns, and trends from massive datasets. It's a cornerstone of data science, empowering organizations to make informed decisions based on insights gleaned from raw data.

Data mining finds applications in a wide range of domains, including:

- Customer Relationship Management (CRM): Understanding customer behavior, preferences, and churn factors to personalize marketing campaigns, improve customer retention, and optimize product offerings.
- Fraud Detection: Identifying anomalies in transactions or credit card usage patterns to prevent fraudulent activity and protect financial systems.
- Recommendation Systems: Recommending products, services, or content to users based on their past purchases, browsing history, and similar user profiles.
- Market Research: Analyzing market trends, competitor strategies, and customer demographics to gain a competitive edge.
- Scientific Discovery: Uncovering hidden relationships and patterns in scientific data to advance research in areas like astronomy, medicine, and biology.

Steps of data mining:

1. **Data Integration:**
    a. Goal: Combine data from disparate sources (databases, spreadsheets, social media feeds) into a unified, consistent format.

    b. Techniques: Data warehousing, entity resolution, data cleansing.

2. **Data Selection:**
    a. Goal: Select a relevant subset of data that aligns with the specific business problem or research question being addressed. Focusing on a targeted dataset improves processing efficiency and reduces computational costs.
    b. Techniques: Statistical sampling, domain knowledge-based selection.
3. **Data Cleaning:**
    a. Goal: Address data quality issues such as missing values, inconsistencies, and outliers. This step ensures the accuracy and reliability of the results.
    b. Techniques: Imputation techniques for missing values, outlier detection and correction, data standardization.
4. **Data Transformation**:
    a. Goal: Prepare the data for mining algorithms. This might involve feature scaling or engineering new features from existing ones to enhance the model's learning ability.
    b. Techniques: normalization, feature selection, dimensionality reduction.
5. **Data Mining:**
    a. Goal: Apply data mining algorithms to uncover patterns and relationships within the data. Common algorithms include **classification** (categorizing data points), **regression** (predicting continuous values), **clustering** (grouping similar data points), and association rule learning (identifying frequently occurring itemsets).
    b. Techniques: Classification trees, decision trees, support vector machines, neural networks, k-means clustering, Apriori algorithm.
6. **Pattern Evaluation:**
    a. Goal: Assess the discovered patterns for validity, significance, and actionable insights. This ensures that the patterns are not merely coincidental and can be used to make informed decisions.
    b. Techniques: Statistical tests, visualization techniques, domain expert review.
7. **Knowledge Representation and Presentation:**
    a. Goal: Communicate the extracted knowledge effectively to stakeholders in a clear, concise, and actionable format. This might involve creating data visualizations, reports, or interactive dashboards.
    b. Techniques: Data visualization tools (e.g., bar charts, scatter plots), reports, dashboards, presentations.

Examples to Illustrate the Power of Data Mining

- Retail: Analyzing customer purchase history to identify buying patterns and recommend complementary products for upselling.
- Finance: Detecting fraudulent credit card transactions by spotting anomalies in spending patterns.
- Healthcare: Predicting patient readmission risks using health records data to provide preventive care and reduce healthcare costs.
- Telecommunications: Identifying churn factors (reasons why customers leave) to design customer retention strategies and improve customer satisfaction.

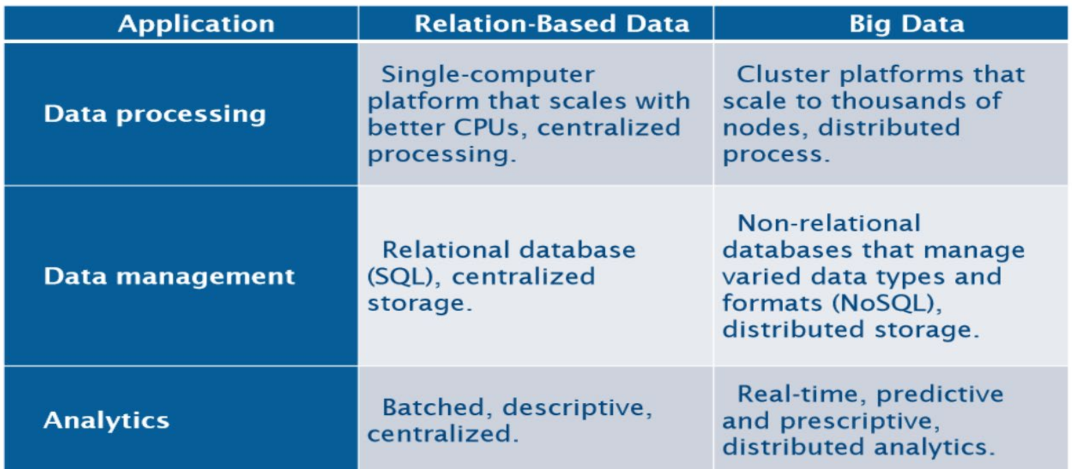
Big DATA

Big data refers to massive, complex datasets that traditional data processing tools struggle to handle. It's characterized by five key dimensions:

- Volume: The sheer amount of data generated from various sources, often growing exponentially.
- Velocity: The speed at which data is created and needs to be processed, analyzed, or acted upon. Real-time or near-real-time processing is crucial.
- Variety: The diverse nature of data, encompassing structured (databases), semi-structured (logs, emails), and unstructured (social media, text) formats.
- Value: Extracting meaningful insights and patterns from the data to drive informed decision-making, improve efficiency, and create new opportunities.
- Veracity: Ensuring the accuracy, reliability, and completeness of the data to avoid misleading results.

Sources of data in Big Data

- Social Media
- Online cloud platforms
- Internet of things
- Online Web pages
- Search Engine Data
- Stock Exchange Data



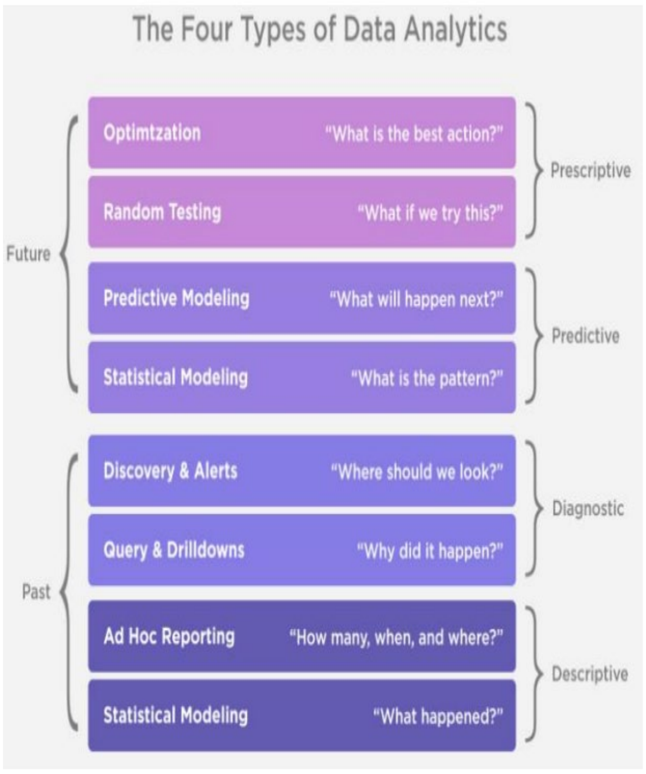
Application	Relation-Based Data	Big Data
Data processing	Single-computer platform that scales with better CPUs, centralized processing.	Cluster platforms that scale to thousands of nodes, distributed process.
Data management	Relational database (SQL), centralized storage.	Non-relational databases that manage varied data types and formats (NoSQL), distributed storage.
Analytics	Batched, descriptive, centralized.	Real-time, predictive and prescriptive, distributed analytics.

Big Data vs. Data Science

Feature	Big Data	Data Science
Focus	Managing and processing large datasets	Extracting knowledge and insights from data
Data Size	Primarily deals with **very large** datasets	Can handle **various data sizes**, including big data

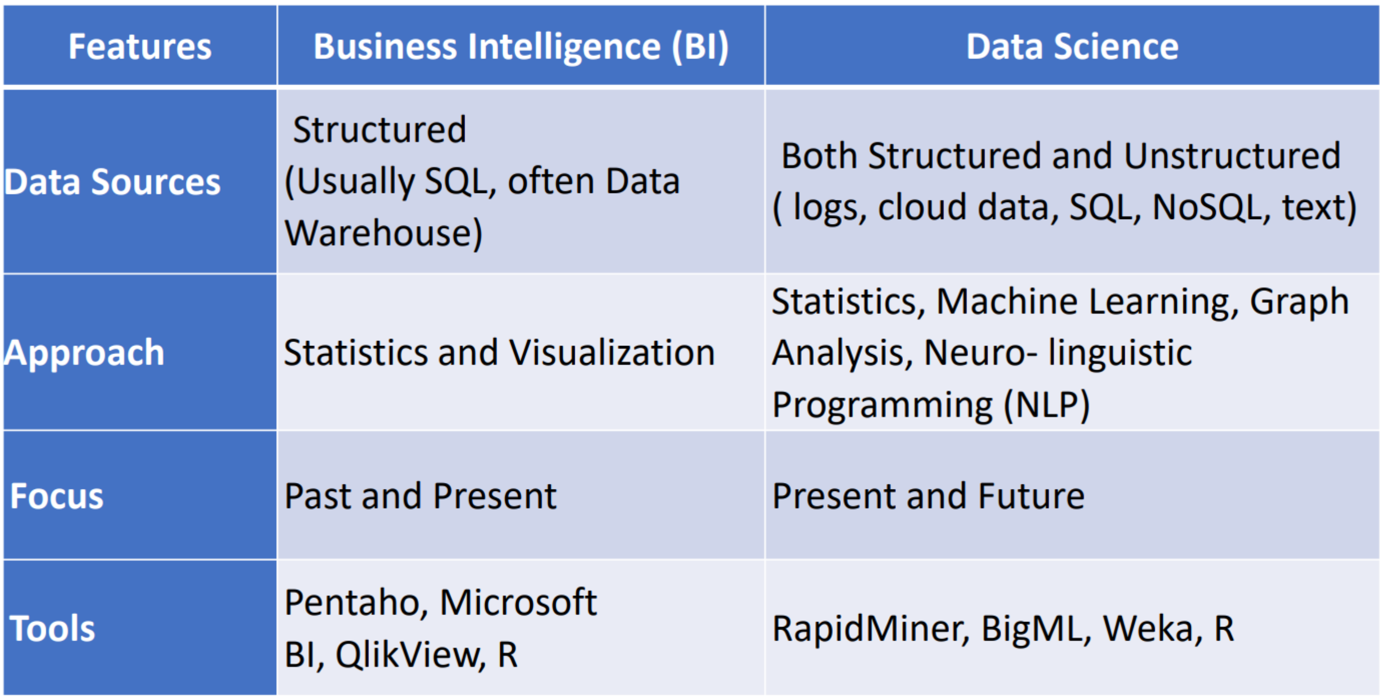
Skills	Data engineering, distributed systems, storage	Statistics, machine learning, programming
Tools	Hadoop, Spark, NoSQL databases	Python, R, SQL, machine learning libraries
Goal	Make massive datasets usable and accessible	Uncover patterns, trends, and actionable insights
Output	Cleaned, organized, and accessible data	Models, predictions, visualizations, reports
Applications	Real-time analytics, fraud detection, log analysis	Customer segmentation, product recommendation, risk assessment

Types of analysis


The Four Types of Data Analytics
Optimtzation "What is the best action?"
Random Testing "What if we try this?"
Predictive Modeling "What will happen next?"
Statistical Modeling "What is the pattern?"
Discovery & Alerts "Where should we look?"
Query & Drilldowns "Why did it happen?"
Ad Hoc Reporting "How many, when, and where?"
Statistical Modeling "What happened?"
Future
Past
Prescriptive
Predictive
Diagnostic
Descriptive

1. **Descriptive Analytics: Understanding What Happened**
    a. Focus: Summarises past data to provide a clear picture of what has transpired.
    b. Techniques: Uses basic statistics (averages, medians, frequencies), data visualisation (charts, graphs).
    c. Example: A retail store analyses its sales data for the past year, identifying top-selling products, seasonal trends, and customer demographics.
2. **Diagnostic Analytics: Uncovering the "Why"**
    a. Focus: Drills down into past data to pinpoint the root causes of events or trends.
    b. Techniques: Data mining, correlation analysis, anomaly detection.
    c. Example: A hospital examines patient readmission rates, identifying factors like specific diagnoses or lack of follow-up care that contribute to readmissions.
3. **Predictive Analytics: Forecasting the Future**
    a. Focus: Leverages historical data and statistical models to predict future outcomes or trends.
    b. Techniques: Regression analysis, machine learning algorithms, forecasting models.
    c. Example: A bank assesses a loan applicant's creditworthiness using a model trained on past loan repayment data, predicting the probability of on-time payments for the new applicant.
4. **Prescriptive Analytics: Taking Action**
    a. Focus: Goes beyond prediction to recommend optimal courses of action based on anticipated outcomes.
    b. Techniques: Optimization algorithms, simulation modelling, decision rules.

c. Example: A ride-sharing service uses real-time traffic data and predictive analytics to suggest the most efficient route for drivers, optimising passenger wait times and driver earnings.



Features	Business Intelligence (BI)	Data Science
Data Sources	Structured (Usually SQL, often Data Warehouse)	Both Structured and Unstructured (logs, cloud data, SQL, NoSQL, text)
Approach	Statistics and Visualization	Statistics, Machine Learning, Graph Analysis, Neuro- linguistic Programming (NLP)
Focus	Past and Present	Present and Future
Tools	Pentaho, Microsoft BI, QlikView, R	RapidMiner, BigML, Weka, R

Lifecycle of Data Science

- Phase 1—Discovery various specifications, requirements, priorities , budget, business problem and hypothesis.
- Phase 2—Data preparation loading, cleaning, transformation, and visualization
- Phase 3—Model planning methods and techniques to draw relationship between variables
- Phase 4—Model building develop datasets for training and testing purposes; classify them
- Phase 5—Operationalize reports, briefings, code and technical documents
- Phase 6—Communicate results you identify all the key findings, communicate to the stakeholders and determine if the results of the project are a success or a failure based on the criteria developed in Phase

IDE

Features of IDE

- Code editor - text editor - highlighting syntax with visual cues
- Compiler /interpreter - human-readable code into machine-specific code
- Debugger – to point out the locations of bugs or errors
- Build automation tools – execution is possible via PLAY
- Version Control – to bring clarity to the development of the software
- Code snippets - to accomplish a single task and to reduce redundant work
- Code navigation- helps to analyse the code
- Extensions and Plugins - to extend the functionality of the IDEs with respect to specific programming languages

Why user ide / advantages

- Productivity: increase due to combining activities like editing code, building executables, debugging, and testing , reduced time

- Code Quality: built-in tools with GUI, no need to switch between apps while developing - help in Syntax highlighting - code analysis
- Integrated Environment: pre-built for new apps – manual configuration is not needed
- Customizability: custom color schemes keyboard shortcuts, choosing unique layouts, different plugins, and add-ons;

Exploratory Data Analysis (EDA) Techniques

It involves investigating, summarizing, and visualizing data to understand its characteristics, identify patterns, and uncover potential relationships between variables.
This process helps data scientists gain insights into the data before diving into more complex modeling or analysis techniques.

Data Cleaning

- Data cleansing
- Process of removing inaccurate records
- Removing dirty records
- Non-relevant part of data
- Unfinished data
- Useless data
- Errors may occur while collecting data
- Transforming data
- Analyzing
- Submission of draft

data scientists spend 80% of their time cleaning and manipulating data and only 20% of their time actually analyzing it.
Techniques for handling these issues include:

- Missing Values
- Inconsistencies
- Outliers

Techniques :

- Descriptive Statistics:Calculating summary statistics for numerical variables
- Central Tendency: Mean, median, mode to understand the "center" of the data distribution.
- Spread: Standard deviation, variance, range to quantify how spread out the data is.
- Distribution: Measures like skewness and kurtosis to assess if the data is symmetrical or skewed (lopsided) and how peaked or flat the distribution is compared to a normal distribution.
- Calculating frequency distributions for categorical variables to understand the distribution of categories and identify potential biases.

Data Visualization:

the art of using visual elements to communicate data clearly and effectively. It's a crucial tool in data science, transforming raw data into understandable charts, graphs, and other visuals.
Effective data visualization helps in:

- Identifying patterns and trends in data.
- Communicating insights to both technical and non-technical audiences.
- Supporting data analysis and storytelling.

When creating data visualizations, it's important to choose the right technique for the type of data you have and the message you want to convey. Here are some common data visualization techniques:

- Histograms: To show the distribution of a single continuous variable.
- Scatter Plots: To explore relationships between two continuous variables.
- Bar Charts: To compare categorical variables or show frequencies within categories.
- Line Charts: To show trends over time or continuous sequences.
- Pie Charts: To represent proportions of a whole for categorical data (use with caution due to limitations in human perception of pie chart slices).
- Boxplots: To visualize the distribution of a numerical variable across different categories.
- Heatmaps: To represent correlations between multiple variables.

Data Sources

1. Relational Databases:

Structured data storage organized in tables with rows and columns. Each table represents a specific entity and rows represent individual records within that entity. Columns represent attributes of those records

Access: Structured Query Language (SQL) is used to retrieve, manipulate, and manage data stored in relational databases.

Advantages:

- Efficient storage and retrieval of structured data.
- Strong data integrity through constraints and relationships between tables.
- Widely used and familiar to many data professionals.

Disadvantages:

- Less flexible for unstructured data (e.g., text, images).
- Can be complex to manage for very large datasets.

2. Web/API Access:

Application Programming Interfaces (APIs) provide programmatic access to data from web services offered by various organizations. They allow data exchange between applications using defined protocols and formats

Access: Programming languages like Python or R can be used with libraries to interact with APIs and retrieve data.

Advantages:

- Access to a vast amount of data from diverse sources.
- Real-time or near real-time data access for certain APIs.
- Automates data retrieval, streamlining data collection.

Disadvantages:

- Reliance on the availability and stability of the API.
- Data formats and access terms may vary across APIs.
- Potential authentication requirements for some APIs.

3. Streaming Data:

Continuous flow of data generated in real-time or near real-time from various sources like social media feeds, sensor readings, or financial transactions.

Access: Specialized tools and frameworks are used to capture, process, and analyze streaming data. Apache Kafka and Apache Flink are common examples.

Advantages:

- Enables real-time insights and analytics.
- Useful for monitoring and anomaly detection applications.

Disadvantages:

- High volume of data can be challenging to manage and store.
- Requires specialized infrastructure and expertise for handling.
- May require real-time processing techniques for immediate analysis.

Data Collection

Data collection is the first step in any data science project. It involves gathering the raw data that will be used for analysis and modeling.

Methods of Data Collection:

1.Sampling

- Random Sampling: Each member of the population has an equal chance of being selected. This is the gold standard for ensuring unbiased data and generalizable results.
- Techniques include:
    - Simple Random Sampling: Every individual has an equal chance of being chosen, often achieved through random number generation.
    - Systematic Random Sampling: The population is ordered in some way, and a random starting point is chosen. Then, every nth individual is selected.
    - Stratified Random Sampling: The population is divided into subgroups (strata) based on relevant characteristics. Random sampling is then performed within each stratum to ensure representation of all subgroups.

- Non-Random Sampling: These techniques can be quicker or cheaper but may introduce bias:
- Convenience Sampling: Selecting readily available individuals, often leading to biased representation.
- Judgment Sampling: The researcher selects individuals based on their judgment of who is most appropriate, potentially introducing subjectivity.
- Quota Sampling: Setting quotas for different subgroups within the sample to ensure representation, but may not be truly random.

Impact on Data Visualization, Modeling, and Generalizability:

- Data Visualization: Sampling bias can lead to misleading visualizations. A well-chosen sample should represent the population accurately to ensure visualizations reflect true patterns and trends.
- Modeling: Biased samples can lead to models that perform well on the specific data used but fail to generalize to the broader population. Random sampling techniques help mitigate this risk.
- Generalizability: The generalizability of results refers to how well your findings from a sample apply to the entire population. Random sampling techniques increase the likelihood that your results can be generalized to the population you're interested in.

2**. Study Design** (Observational vs. Experimental)

Observational Studies: Analyze existing data without manipulating variables. They can identify associations between variables but cannot definitively establish cause-and-effect relationships. Here are some common types of observational studies:

Cohort Studies: Follow a group of individuals (the cohort) over time, often for years or decades. Researchers observe how exposure to a factor (e.g., smoking) affects outcomes (e.g., lung cancer) within the cohort.
Example: A study following a group of healthcare workers from different departments (exposed/not exposed to radiation) over 20 years to investigate the potential link between radiation exposure and cancer rates.

Case-Control Studies: Compare individuals with a specific outcome (cases) to those without it (controls). Researchers then examine past exposures or characteristics to identify potential risk factors associated with the outcome.
Example: A study comparing women with breast cancer (cases) to women without breast cancer (controls) to investigate past lifestyle habits and potential risk factors.

Cross-Sectional Studies: Examine relationships between variables at a single point in time. They can identify associations but cannot determine the direction of causality (which variable came first).
Example: A survey at a grocery store to investigate the relationship between customer age and preferred type of milk (whole, low-fat, etc.).

Experimental Studies: Manipulate variables to observe their effect on an outcome. They can establish cause-and-effect relationships by controlling for other factors that might influence the outcome.

Randomized Controlled Trials (RCTs): Considered the gold standard for establishing causality. Participants are randomly assigned to either a treatment group (receives the intervention) or a control group (receives a placebo or no intervention). Researchers then measure the outcome to see if the treatment has a causal effect.
Example: A clinical trial where participants with high blood pressure are randomly assigned to receive a new medication (treatment group) or a placebo (control group) to assess the medication's effectiveness in lowering blood pressure.

Pre-test/Post-test Designs: Measure an outcome before and after an intervention to assess its impact. However, external factors that occur between the pre-test and post-test might influence the results, making it challenging to establish a clear cause-and-effect relationship.
Example: A pre-test/post-test design might be used to evaluate the effectiveness of a new educational program. Students' test scores are measured before and after the program to assess learning gains. However, other factors like individual study habits could also influence the post-test scores.

Impact on Data Visualization, Modeling, and Generalizability:
The study design, whether observational or experimental, affects how you interpret data and the conclusions you can draw:

- Data Visualization: Observational studies may reveal correlations between variables, but visualizations cannot definitively establish cause-and-effect relationships.
- Modeling: Models built on observational data can predict future outcomes based on existing relationships, but they cannot isolate the impact of specific variables due to potential confounding factors.

- Generalizability: The generalizability of results depends on the study design. Experimental studies can provide stronger evidence of causality, but their generalizability can be limited if the experiment doesn't reflect real-world conditions.

Data Cleaning/Extraction

Raw data often contains errors, inconsistencies, and missing values. Data cleaning and extraction involve preparing the data for analysis by identifying and addressing these issues.
Data Cleaning Techniques:

- Identifying Issues: Look for missing values, outliers, inconsistencies in formatting or coding, and potential errors.
- Missing Value Imputation: Depending on the data type and distribution, techniques like mean/median imputation, k-Nearest Neighbors, or more sophisticated methods can be used to fill in missing values.
- Handling Outliers: Decide whether to keep, winsorize (cap outliers to a certain threshold), or remove outliers based on domain knowledge and potential impact on analysis.
- Data Transformation: Techniques like normalization (scaling features) or encoding categorical variables might be applied for specific analysis requirements.

Data Extraction:

For large datasets or complex data sources, extraction techniques are used to select and retrieve specific data subsets relevant to the analysis. Tools like SQL queries or APIs can be used for extraction.
Clean data is crucial for reliable analysis and accurate model building. Dirty data can lead to misleading results and hinder the success of your data science project.

Data Analysis & Modeling

Data analysis and modeling involve using statistical techniques and machine learning algorithms to uncover patterns, trends, and relationships within the data.
Data Analysis Techniques:

- Exploratory Data Analysis (EDA): This initial analysis involves summarizing, visualizing, and exploring the data to gain initial insights, identify patterns, and understand data distribution.
- Descriptive Statistics: Calculations like mean, median, standard deviation, and frequency distributions summarize the data and provide a foundational understanding of its central tendency, spread, and distribution.
- Correlation Analysis: Measures the strength and direction of linear relationships between numerical variables, helping identify potential relationships for further exploration.

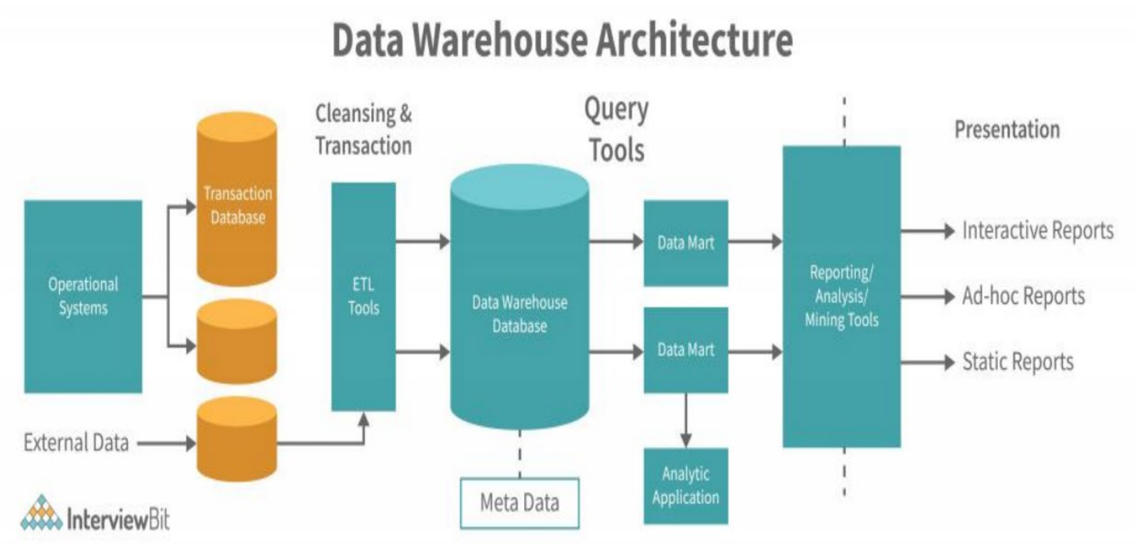
Data Modeling Techniques:

- Model Selection: Choose the appropriate modeling technique based on the data type (numerical, categorical) and the desired outcome (prediction, classification, etc.). Common models include linear regression, logistic regression, decision trees, random forests, etc.
- Model Training: Train the model on a portion of the data, allowing it to learn the patterns and relationships within the data.
- Model Evaluation: Evaluate the model's performance on a separate test dataset to assess its generalizability and effectiveness in making predictions or classifications on unseen data.

- Data analysis and modeling are often iterative processes. Based on initial results, you may refine your analysis techniques, feature selection, or model training approach to improve model performance.

Data Warehouses

Data warehouse
• A Data Warehouse is separate from DBMS, it stores a huge amount of
data
• collected from multiple sources like files, DBMS ,….
• to produce statistical results
• Strategic decision making
Features of Data Warehousing
• Centralized Data Repository
• Data Integration
• Historical Data Storage
• Query and Analysis
• Data Transformation
• Data Mining
• Data Security – backup, encryption…



Benefits of warehouse
- Intelligent Decision-Making: With centralized data in warehouses, decisions may
- be made more quickly and intelligently.
- Business Intelligence: Provides strong operational insights through business intelligence.
- Historical Analysis: Predictions and trend analysis are made easier by storing past data.
- Data Quality: Guarantees data quality and consistency for trustworthy reporting.
- Scalability: Capable of managing massive data volumes and expanding to meet changing requirements.

- Effective Queries: Fast and effective data retrieval is made possible by an optimized structure.
- Cost reductions: Data warehousing can result in cost savings over time by reducing data management procedures and increasing overall efficiency, even when there are setup costs initially.
- Data security: Data warehouses employ security protocols to safeguard confidential information, guaranteeing that only authorized personnel are granted access to certain data.

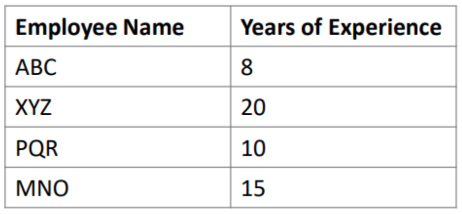
Disadvantages of Data Warehousing
- Cost: Building a data warehouse can be expensive, requiring significant investments in hardware, software, and personnel.
- Complexity: Data warehousing can be complex, and businesses may need to hire specialized personnel to manage the system.
- Time-consuming: Building a data warehouse can take a significant amount of time, requiring businesses to be patient and committed to the process.
- Data integration challenges: Data from different sources can be challenging to integrate, requiring significant effort to ensure consistency and accuracy.
- Data security: Data warehousing can pose data security risks, and businesses must take measures to protect sensitive data from unauthorized access or breaches.

Data Preprocessing

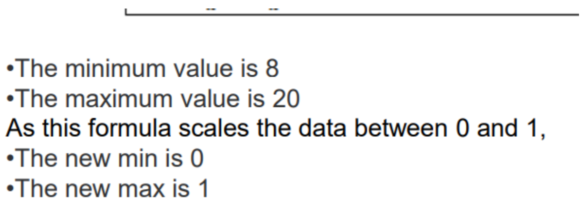
Data Normalization
- Normalization is used to scale the data of an attribute so that it falls in a smaller range, such as -1.0 to 1.0 or 0.0 to 1.0.
- It is generally useful for classification algorithms.

Advantages of Data Normalization in Data
Mining
• the application of data mining algorithms becomes easier
• the data mining algorithms get more effective and efficient
• the data is converted in to the format that everyone can get their
heads around
• the data can be extracted from databases faster
• it is possible to analyze the data in a specific manner

Min-max normalization
A technique in data science used to scale numeric data points into a fixed range, typically between 0 and 1. This is accomplished through a linear transformation, which adjusts the values based on the original dataset's minimum and maximum values.
Formula = v′ = (v - min) / (max - min) * (new_max - new_min) + new_min
v′ = (10 - 8) / (20 - 8) * (1 - 0) + 0
v′ = 0.16 //for picc below sum can come



Employee Name	Years of Experience
ABC	8
XYZ	20
PQR	10
MNO	15



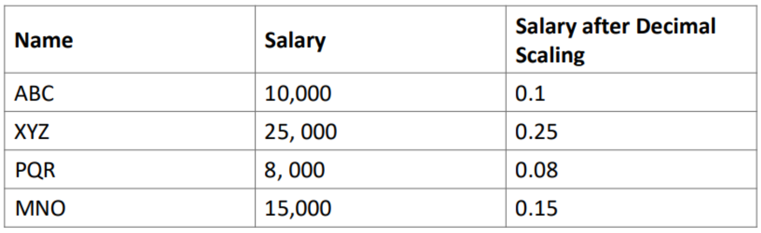
•The minimum value is 8
•The maximum value is 20
As this formula scales the data between 0 and 1,
•The new min is 0
•The new max is 1

Decimal Scaling

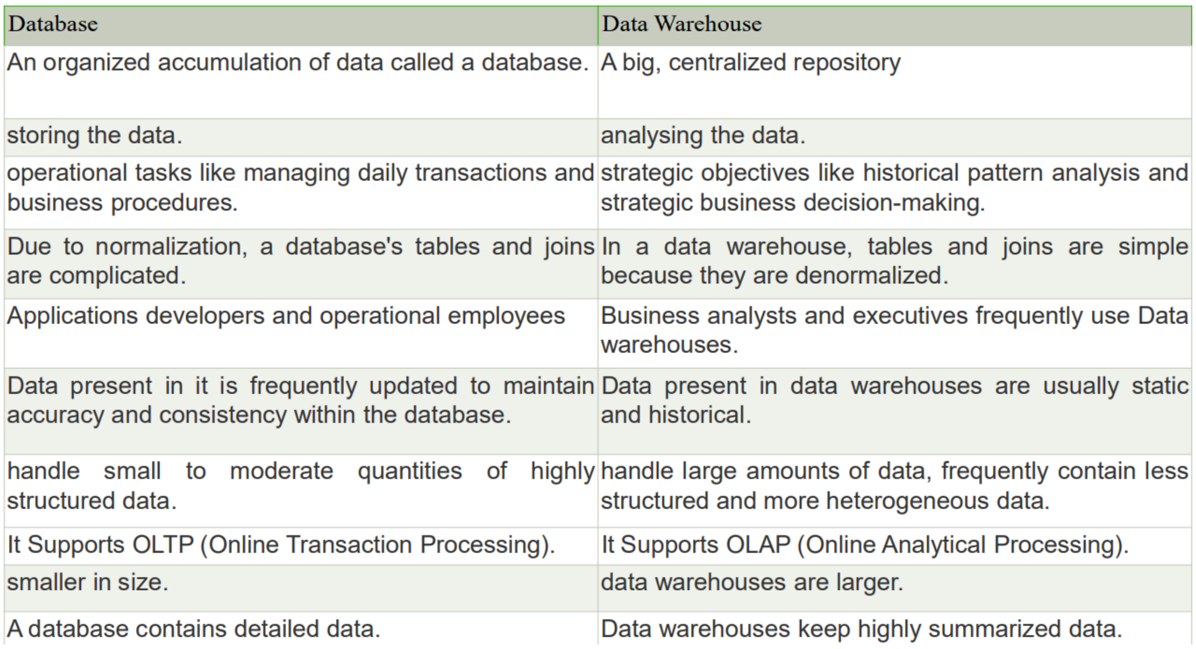
- Decimal scaling scales the data points relative to the maximum absolute value, so it doesn't necessarily guarantee a specific range like min-max normalization.
- It's a simpler approach compared to min-max normalization and can be useful when you know the data falls within a specific range.
- However, it can be sensitive to outliers, which can significantly alter the scaling factor and lead to unintended consequences.

$v' = v / (10^j)$
v represents the original data value you want to normalize.
v′ represents the normalized value.
j represents the number of decimal places in the maximum absolute value.

The maximum absolute value is ₹25,000.
There are four decimal places (0.00025).
Apply the formula:
$v' = ₹10,000 / (10^4) = ₹0.10$



Name	Salary	Salary after Decimal Scaling
ABC	10,000	0.1
XYZ	25, 000	0.25
PQR	8, 000	0.08
MNO	15,000	0.15



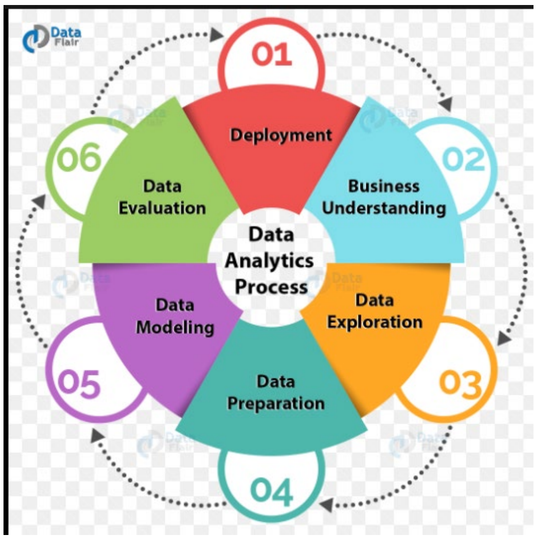
Database	Data Warehouse
An organized accumulation of data called a database.	A big, centralized repository
storing the data.	analysing the data.
operational tasks like managing daily transactions and business procedures.	strategic objectives like historical pattern analysis and strategic business decision-making.
Due to normalization, a database's tables and joins are complicated.	In a data warehouse, tables and joins are simple because they are denormalized.
Applications developers and operational employees	Business analysts and executives frequently use Data warehouses.
Data present in it is frequently updated to maintain accuracy and consistency within the database.	Data present in data warehouses are usually static and historical.
handle small to moderate quantities of highly structured data.	handle large amounts of data, frequently contain less structured and more heterogeneous data.
It Supports OLTP (Online Transaction Processing).	It Supports OLAP (Online Analytical Processing).
smaller in size.	data warehouses are larger.
A database contains detailed data.	Data warehouses keep highly summarized data.

Basic terms
• Independent variables: Data that can be controlled directly.
• Dependent variables: Data that cannot be controlled directly.
Experiment 1: You want to figure out which brand of microwave popcorn pops the most kernels so you can
get the most value for your money. You test different brands of popcorn to see which bag pops the most
popcorn kernels.

•Independent Variable: Brand of popcorn bag (It’s the independent variable because you are actually
deciding the popcorn bag brands)
•Dependent Variable: Number of kernels popped (This is the dependent variable because it's what you
measure for each popcorn brand)
Experiment 2: You want to see which type of fertilizer helps plants grow fastest, so you add a different brand
of fertilizer to each plant and see how tall they grow.
•Independent Variable: Type of fertilizer given to the plant
•Dependent Variable: Plant height

Data analysis diagram

- Data Collection: Gather the data required for your analysis.
- Data Evaluation: Assess the quality and suitability of the collected data.
- Data Understanding: Gain insights into the data's structure and characteristics.
- Data Modeling: Develop a model to represent the data and extract patterns.
- Data Preparation: Clean and pre-process the data for analysis.
- Data Analysis: Perform statistical or machine learning techniques to analyze the data.
- Deployment: Communicate the results and insights from your analysis.



Unit 2

Query languages

Query languages are specialized programming languages designed to interact with data storage systems and retrieve specific information. They act as an intermediary between the user and the database, allowing the user to express your data retrieval needs in a human-readable format.

Different query languages are used for different types of databases:

- SQL (Structured Query Language): The most widely used query language for relational databases. It allows you to perform various operations like selecting specific data, filtering based on conditions, joining tables, and sorting results.

- NoSQL Query Languages: Used for querying non-relational databases like document stores (e.g., MongoDB) or key-value stores (e.g., Redis). These languages vary depending on the specific database system but often share a focus on flexibility and ease of use for unstructured or semi-structured data.
- XPath and XQuery: Primarily used for querying XML data. XPath is a path-based language for navigating the structure of an XML document and locating specific elements. XQuery is a more advanced language that allows for complex queries and data manipulation on XML data.

operations:

- Select: This operation specifies which data columns or attributes you want to retrieve from the database.
- Filter (Where): Allows you to filter the retrieved data based on specific conditions. For example, you could filter a table of customers to only show those located in a particular city.
- Join: Used to combine data from multiple tables in a relational database based on a shared field. This allows you to analyze relationships between data stored in separate tables.
- Delete: to delete specific data from rows or delete entire rows and columns
- Update: to update specific row column values
- Sort: Arranges the retrieved data in a specific order (ascending or descending) based on a chosen column.
- Aggregate Functions: Perform calculations on entire datasets or groups of data. Examples include functions to calculate sum, average, minimum, maximum, and count.
- Group By: Groups data rows based on a shared value in a specific column. This allows you to perform aggregate functions on groups of data.

Structured Data Storage (Schema-based Systems)

Structured data is highly organized and follows a predefined format, typically stored in relational databases. Relational databases are the workhorses of data storage for many organizations. Here's a breakdown of how they work:

- Schema: The foundation of a relational database is its schema, which defines the structure of the data. It specifies the tables, columns (attributes), and data types for each column. This predefined structure ensures consistency and memudahkan data retrieval and manipulation. ([ease of use] means "makes it easier" in Indonesian)
- Tables: Data is organized into tables, where each row represents a record (individual entity) and each column represents an attribute (characteristic) of that record. For example, a customer table might have columns for customer ID, name, address, and email.
- Relationships: Relational databases excel at establishing relationships between tables. These relationships are often defined through foreign keys, which link data points across tables. This allows you to efficiently retrieve and analyze data from multiple tables simultaneously.

Benefits of Structured Data Storage:

- Efficient Storage and Retrieval: Schema-based organization allows for fast and efficient data retrieval using SQL queries.
- Data Integrity: Relational databases enforce data integrity through constraints and data types, helping to maintain data consistency and accuracy.

- Strong Foundation for Analysis: Structured data serves as a strong foundation for data analysis and reporting due to its organization and ease of querying.

Limitations of Structured Data Storage:

- Less Flexible for Unstructured Data: Relational databases struggle to store and manage unstructured data like text, images, or videos.
- Complexity for Large Datasets: Managing and scaling relational databases for very large datasets can become complex and resource-intensive.

Semi-structured Data Storage

Semi-structured data sits between structured and unstructured data. It has some internal organization but lacks a rigid schema like a relational database. Here are some common storage formats for semi-structured data:

- XML (Extensible Markup Language): Uses tags and attributes to define the structure of data. XML provides a standard format for data exchange and can be easily parsed by various applications.
- JSON (JavaScript Object Notation): A lightweight format that uses key-value pairs to represent data. JSON is popular for data exchange due to its human-readable syntax and ease of use in web applications.

Benefits of Semi-structured Data Storage:

- Flexibility: Semi-structured data offers more flexibility than structured data, allowing for variations in data format within a category.
- Scalability: Semi-structured data storage can be more scalable than relational databases for very large datasets.
- Ease of Integration: Formats like JSON are often easier to integrate with web applications and APIs.

Limitations of Semi-structured Data Storage:

- Complexity for Complex Queries: Querying and analyzing semi-structured data compared to relational databases can be more complex due to the lack of a predefined schema.
- Data Validation: Maintaining data consistency can be a challenge due to the looser structure of semi-structured data.

Unstructured Data Storage

Unstructured data lacks a rigid format and encompasses a wide variety of data types like text documents, images, audio, video, social media posts, and sensor data. Here's how unstructured data is typically stored and managed:

- File Systems: Traditional file systems are used to store individual files, but managing and analyzing large volumes of unstructured data in this way can be cumbersome.
- Data Lakes: These are large repositories that store vast amounts of raw, unstructured data in its native format. Data lakes provide flexibility for storing diverse data types but often require additional processing before analysis.
- NoSQL Databases: These are non-relational databases designed for storing and managing large volumes of unstructured or semi-structured data. NoSQL databases offer scalability and flexibility for handling diverse data formats but may require specialized querying techniques compared to SQL.

Benefits of Unstructured Data Storage:

- Versatility: Unstructured data storage allows for storing a wide variety of data types, essential for capturing the richness of real-world information.
- Scalability: NoSQL databases and data lakes can efficiently scale to handle massive volumes of unstructured data.

Limitations of Unstructured Data Storage:

- Complexity of Analysis: Unstructured data often requires additional processing and transformation before it can be analyzed effectively.
- Storage Costs: Storing large amounts of unstructured data can come with significant storage costs.

Ethical Considerations for Authorization and Authentication

Granting access to data raises ethical considerations. key principles:

- Least Privilege: Users should only be granted the minimum level of access needed to perform their job duties. This principle minimizes the potential damage caused by unauthorized access or human error.
- Data Privacy: Organizations have a responsibility to protect user privacy and only collect and store data that is necessary for legitimate business purposes. Users should be informed about how their data is being used and have control over their data privacy settings.
- Accountability and Transparency: Clear policies and procedures should define who can access data, for what purposes, and the consequences of misuse. Transparency about data security practices builds trust with users.
- Security Awareness Training: Educating users about cybersecurity best practices, like strong password hygiene and recognizing phishing attempts, is crucial for maintaining a strong security posture.

Data analysis and modeling

Data analysis starts with a clear understanding of the questions you're trying to answer and the data you have available. This section explores problem formulation and Exploratory Data Analysis (EDA), essential initial steps in your data analysis journey.

Problem Formulation

The foundation of any successful data analysis project is a well-defined problem statement. This clarifies your goals and guides your data exploration and modeling choices. Here's how to formulate a strong problem statement:

- Identify the Business Objective: Start by understanding the broader business context and the specific objective your analysis aims to achieve. Is it to improve customer satisfaction, predict sales trends, or identify risk factors?
- Translate to a Data-driven Question: Phrases your objective as a question that data analysis can answer. For instance, "Can we identify factors influencing customer churn?" or "How well can we predict product sales based on past data?"
- Define Success Metrics: Determine how you'll measure the success of your analysis. This could be metrics like increased customer retention rate, improved sales forecasting accuracy, or a reduction in fraud detection time.

Exploratory Data Analysis (EDA)

Once you have a well-defined problem, EDA comes into play. EDA is the process of getting acquainted with your data, understanding its structure, and uncovering initial insights. It's like detective work, where you explore the data to find clues and patterns that might be relevant to your analysis question.
Here are some key EDA techniques:

- Data Cleaning and Preprocessing: This initial stage involves checking for missing values, inconsistencies, and outliers in your data. Cleaning and preprocessing ensure the quality of your data and the reliability of your analysis results.
- Understanding Data Distribution: Techniques like histograms, boxplots, and descriptive statistics help you visualize and understand the distribution of your data (e.g., Is a variable spread out evenly or skewed towards one side?).
- Identifying Relationships: Techniques like scatter plots and correlation analysis can reveal potential relationships between variables in your data. These relationships might be crucial for building your models.
- Data Visualization: Creating various charts and graphs like bar charts, pie charts, and heatmaps can provide visual summaries of your data, making it easier to identify patterns and trends.

By effectively conducting EDA, you gain a deeper understanding of your data's strengths and weaknesses. This knowledge is instrumental in choosing the right modeling techniques and interpreting your results accurately.

Introduction to Statistical Inference

Statistical inference is a cornerstone of data analysis. It allows you to use data from a small sample (what you can realistically collect or measure) to draw conclusions about a larger population (the entire group you're interested in). This section dives into three key concepts: estimation, hypothesis testing, and the scope of inference.

Estimation and Hypothesis Testing

Imagine you want to understand the average height of all students in your university (the population). It's impractical to measure everyone's height, so you collect data from a smaller sample group (e.g., 100 students). Statistical inference helps you bridge the gap between this sample and the entire population.

- Estimation: We often want to estimate population parameters, like the average height (mean) or proportion of students with a certain characteristic. Techniques like confidence intervals provide a range of values within which the true population parameter is likely to fall, with a certain level of confidence (e.g., 95% confidence).
- Hypothesis Testing: Sometimes, we want to test a specific claim about a population parameter. For instance, a hypothesis test could assess if the average student height is greater than 170 cm. Hypothesis testing involves formulating a null hypothesis (usually there's no difference) and an alternative hypothesis (the specific claim you want to test). Based on the sample data, we calculate a test statistic and a p-value to determine if the evidence supports rejecting the null hypothesis.

Bootstrapping as a Resampling Technique:

Bootstrapping is a statistical method that leverages resampling to address these limitations.
Bootstrapping Process:

- Resample with Replacement: The core idea is to create many simulated datasets ("bootstrap samples") by randomly sampling with replacement from your original data. This means a data point can be chosen multiple times in a single bootstrap sample.
- Recalculate Statistic: For each bootstrap sample, you recalculate the statistic you're interested in (e.g., mean, confidence interval). This gives you a distribution of the statistic based on the resampled data.
- Analyze Variability: By analyzing the distribution of the statistic across all bootstrap samples, you gain insights into the variability of your original estimate.

Benefits of Bootstrapping:

- Less Reliant on Assumptions: Bootstrapping is less reliant on strict assumptions about the underlying data distribution compared to some traditional methods.
- Useful for Complex Data: It can be a valuable tool for analyzing complex data or smaller datasets where traditional methods might be less reliable.
- Strength of Conclusions: The distribution of the statistic from bootstrapping helps assess the strength of your conclusions and the confidence you can have in your estimate.

Scope of Inference

The conclusions you draw from your analysis can only be generalized to the population your sample represents. Here are some factors to consider:

- Sampling Method: Random sampling techniques ensure your sample reflects the characteristics of the larger population. Non-random samples can lead to biased results.
- Sample Size: Larger sample sizes generally lead to more reliable estimates and more generalizable conclusions. However, there's always a balance between collecting enough data and feasibility.

GRAPHS

```
import matplotlib.pyplot as plt
import numpy as np

# Sample data
x = np.linspace(0.0, 5.0, 100)
y1 = np.sin(x)
y2 = np.cos(x)
categories = ["A", "B", "C", "D", "E"]
values = [10, 20, 30, 25, 15]

# Line chart
plt.plot(x, y1, label="Sine")
plt.plot(x, y2, label="Cosine")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Line Chart")
plt.legend()
plt.grid(True)
plt.show()
```

```
# Bar chart
plt.bar(categories, values, color='skyblue')
plt.xlabel("Categories")
plt.ylabel("Values")
plt.title("Bar Chart")
plt.xticks(rotation=45)  # Rotate category labels for better readability
plt.tight_layout()
plt.show()

# Scatter plot
plt.scatter(x, y1, color='red', label="Data")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Scatter Plot")
plt.legend()
plt.grid(True)
plt.show()

# Pie chart
plt.pie(values, labels=categories, autopct="%1.1f%%", startangle=140)
plt.title("Pie Chart")
plt.axis("equal")  # Equal aspect ratio ensures a circular pie chart
plt.show()

# Area plot
plt.fill_between(x, y1, color='green', alpha=0.4, label="Area")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Area Plot")
plt.legend()
plt.grid(True)
plt.show()

# Histogram
plt.hist(y1, bins=10, edgecolor='black')
plt.xlabel("Values")
plt.ylabel("Frequency")
plt.title("Histogram")
plt.grid(True)
plt.show()
```

5.graphs why important give example advantages and show data in practical way open ended(feka maaro)
1. Line Chart:
Why it's important: Shows trends or changes over time (continuous data). Useful for visualizing how a variable changes in relation to another variable over time.
Example: Tracking stock prices over a month, temperature variations throughout the day.

Advantages: Simple to interpret, highlights trends and patterns easily.

2. Bar Chart:
Why it's important: Compares categories or discrete data sets. Useful for showing comparisons between different groups or entities.
Example: Customer satisfaction ratings across different products, sales figures for various departments.
Advantages: Easy to understand, effective for comparing categories with clear differences.

3. Scatter Plot:
Why it's important: Reveals relationships between two variables. Useful for identifying correlations or lack thereof between two sets of data.
Example: Relationship between study hours and exam scores, correlation between age and income.
Advantages: Shows potential relationships and outliers, good for exploring data.

4. Pie Chart:
Why it's important: Shows the composition of a whole. Useful for visualizing the proportions of different categories that make up a total value.
Example: Market share distribution among different companies, budget allocation across various spending categories.
Advantages: Easy to understand, good for showing proportions of a whole at a glance (limited to a few categories).

5. Area Plot:
Why it's important: Emphasizes the magnitude of change over time. Useful for visualizing cumulative changes or quantities over time.
Example: Total rainfall throughout a year, sales volume trends over a period.
Advantages: Highlights the overall magnitude of change, good for showing trends with emphasis on area under the curve.

6. Histogram:
Why it's important: Shows the distribution of data. Useful for understanding how data points are spread out and identifying potential outliers.
Example: Distribution of exam scores in a class, income levels in a population.
Advantages: Reveals the shape of the data distribution, helps identify potential skewness or outliers.

Unit 3

Aws

Importance of AWS Datasets:

- Rich Variety: AWS offers a vast collection of datasets across various domains like healthcare, finance, weather, and more. This allows businesses to leverage pre-existing data for analytics and machine learning projects without starting from scratch.
- Scalability and Accessibility: Datasets on AWS are readily available and scalable, enabling businesses to access and analyze massive datasets efficiently.

- Reduced Costs: Using AWS datasets can be more cost-effective than collecting and storing data yourself, especially for large datasets.
- Improved Time-to-Insight: Pre-existing datasets can accelerate the time it takes to gain insights from data analysis.

Why Choose AWS Over Others and Personal Data Collection:

- Security and Compliance: AWS prioritizes data security and offers robust compliance features to meet various industry regulations.
- Reliability and Scalability: AWS infrastructure is known for its reliability and ability to scale to meet growing data needs.
- Managed Services: AWS offers managed services like Amazon Redshift Spectrum that simplify data analysis tasks.
- Focus on Core Business: By utilizing AWS datasets, businesses can focus their resources on core competencies rather than data collection and management.

Personal Data Collection vs. AWS Datasets:

- Cost and Time: Collecting and managing your data can be expensive and time-consuming, especially for large datasets.
- Quality and Bias: Personal data collection might be limited or biased based on your sources.
- Compliance Challenges: Ensuring compliance with data privacy regulations can be complex with personal data collection.

Types of Datasets on AWS:

- GNOMAD: A large collection of human exome and genome sequencing data valuable for genomics research.
- Landsat: Satellite imagery datasets used for environmental monitoring, land-use analysis, and more.
- Multimedia Datasets: Datasets containing audio, video, and image data used for tasks like image recognition and natural language processing.

Introduction to NoSQL

NoSQL, or Not Only SQL, encompasses a wide range of database technologies designed to address the limitations of traditional relational databases. It offers flexibility and scalability for modern data models.

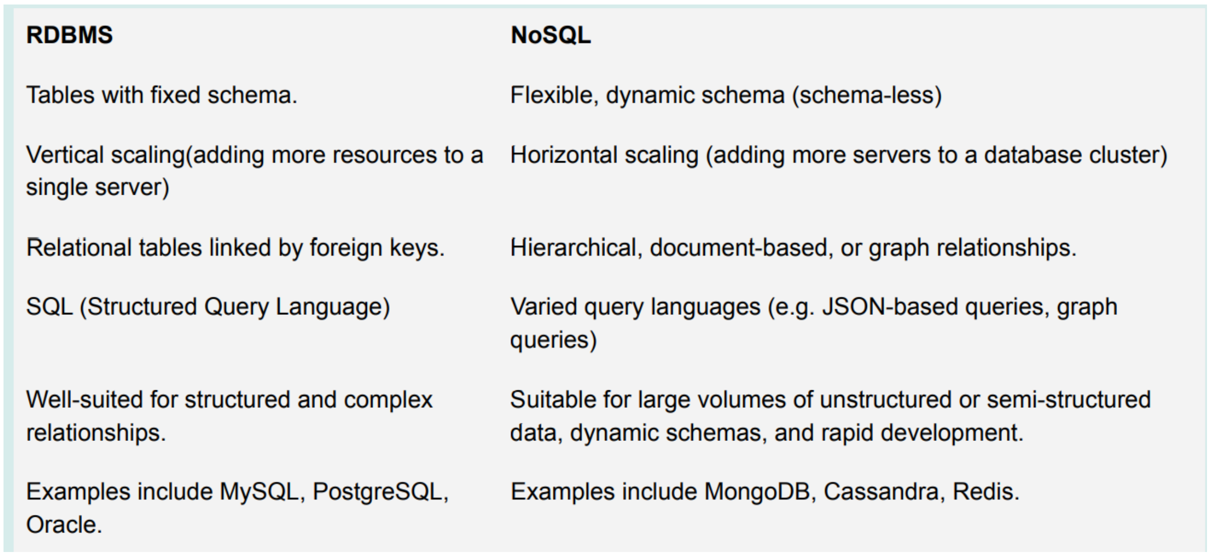
- Flexibility - NoSQL databases are designed to handle unstructured or semi-structured data, which means that they can accommodate dynamic changes to the data model.\
- Scalability - NoSQL databases are highly scalable, which means that they can handle large amounts of data and traffic with ease. This makes them a good fit for applications that need to handle large amounts of data or traffic
- High performance - NoSQL databases are often optimized for specific use cases, providing superior performance for tasks such as real-time analytics, content management, and other scenarios with demanding data access requirements. They can outperform traditional relational databases in certain applications due to their tailored data models and indexing strategies.
- COST EFFECTIVENESS - NoSQL databases are often more cost-effective than traditional relational databases, as they are typically less complex and do not require expensive hardware or software.Their ability to scale horizontally on commodity hardware can result in more efficient resource utilization and reduced infrastructure costs compared to vertically scaling traditional relational databases.

- Agility

Mongo db
MongoDB is a source-available, cross-platform, docu ment-oriented database program. Classified as a NoSQL database product, MongoDB utilizes JSON-like documents with optional schemas
Highlight key features of MongoDB:
- Schema-less
- Document-Oriented
- Scalability: MongoDB provides horizontal scalability, allowing you to scale out by adding more servers to a MongoDB cluster.
- Flexibility: MongoDB is versatile and can be used for a wide range of applications, from small projects to large-scale enterprise systems.
- Unlike relational databases, MongoDB doesn't require a predefined schema. Each document in a collection can have a different structure.
- Dynamic-Schema Does not require fixed column definitions
- It is adaptable to changing requirements.

DOCUMENT-ORIENTED
Data is stored in BSON (Binary JSON) documents. These documents can contain nested structures, arrays, and other complex data types. BSON(Binary JSON)- BSON's binary structure encodes type and length information, which allows it to be traversed much more quickly compared to JSON

MongoDB Data Model
The data model in MongoDB is based on collections and documents:
- Collection: A grouping of MongoDB documents. Collections are equivalent to tables in relational databases.
- Document: A BSON object, similar to a JSON object, that represents a record in MongoDB. Documents can have varying fields and structures.



RDBMS	NoSQL
Tables with fixed schema.	Flexible, dynamic schema (schema-less)
Vertical scaling(adding more resources to a single server)	Horizontal scaling (adding more servers to a database cluster)
Relational tables linked by foreign keys.	Hierarchical, document-based, or graph relationships.
SQL (Structured Query Language)	Varied query languages (e.g. JSON-based queries, graph queries)
Well-suited for structured and complex relationships.	Suitable for large volumes of unstructured or semi-structured data, dynamic schemas, and rapid development.
Examples include MySQL, PostgreSQL, Oracle.	Examples include MongoDB, Cassandra, Redis.

Codes
Create
const mongoose = require('mongoose');

```
const express = require('express');
const connectionString =
'mongodb+srv://aryan:aryanaryu@cluster0.5er5ylb.mongodb.net/test?retryWrites=true&w=m
ajority';
const port = 8080;
const app = express();
(async () => {
  try {
      await mongoose.connect(connectionString, {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    });
    console.log('Connected to MongoDB Atlas');
// idhar tak common for all
    const characterSchema = new mongoose.Schema({
      name: { type: Number, required: true },
      height: String,
      Rollno: Number,
    }, { versionKey: false });
    // Create the Character model with a single line
    const Character = mongoose.model('Character', characterSchema);
```

Retrieve

```
const filter = { name: 'Jean-Luc Picard' };
  doc = await Character.findOne(filter);
  console.log('Roll number from users document where name is ',doc.name,' is : ',
doc.Rollno);
```

Update

```
  const update = { height: '5 foot 12 inches' };
  let doc = await Character.findOneAndUpdate(filter, update, {
    new: true,
    upsert:true,
  });
```

Delete

```
const del_doc=await Character.findOneAndDelete(filter);
console.log("Deleted Document : ",del_doc);
```

CRISP - DM and its Phases

CRISP-DM, which stands for Cross-Industry Standard Process for Data Mining, is a widely used methodology for data mining projects. It provides a structured approach to guide data scientists and analysts through the process of extracting valuable insights from data. The CRISP-DM methodology consists of six main phases:

- Business Understanding: This initial phase focuses on understanding the business objectives and requirements from a data science perspective. It involves identifying the goals of the data science project and aligning them with the business needs.
- Data Understanding: In this phase, data collection is initiated to explore and understand the characteristics of the data. It involves assessing data quality, identifying data sources, and gaining insights into the data to prepare for further analysis.

- Data Preparation: The data preparation phase involves activities to construct the final dataset for modelling tools. This phase ensures that the data is cleaned, transformed, and formatted appropriately to be used in the modelling phase.
- Modelling: The modelling phase involves selecting and applying various data science modelling techniques to build models that meet the business objectives. Different modelling techniques are evaluated to achieve the desired outcomes.
- Evaluation: In this phase, the data science process is evaluated to ensure that the proposed solution meets the business objectives effectively. Validation of the data science solution is crucial before proceeding to the final deployment phase.
- Deployment: The deployment phase involves implementing the data science solution into production. Once the data science solution has passed the development and pilot phases, it is deployed for practical use within the organization.

Ontology

- Ontology, in the context of data science, refers to a formal representation of knowledge within a domain that includes the entities, relationships, and rules governing those entities.
- It provides a structured framework for organizing information and defining the concepts and relationships within a specific area of interest.
- Ontologies are typically used to facilitate data integration, knowledge sharing, and reasoning processes in various applications.

Need of ontology

- Semantic Interoperability: Ontologies enable semantic interoperability by providing a common understanding of data across different systems or domains.
- Data Integration: They help in integrating heterogeneous data sources by standardizing the representation of concepts and relationships.
- Knowledge Sharing: Ontologies facilitate knowledge sharing by defining explicit semantics that can be understood by both humans and machines.
- Automated Reasoning: They support automated reasoning processes by formalising knowledge and enabling logical deductions.
- Data Analysis: Ontologies enhance data analysis by structuring information in a way that allows for more efficient retrieval and analysis.

Example in Action:
Say you have data about movies and actors. An ontology could define:

- Entities: Movies, actors, directors, genres.
- Relationships: Actors star in movies, directors direct movies.
- Rules: An actor can star in multiple movies (one-to-many relationship).

With this ontology, you can:

- Search for all movies starring a specific actor.
- Find movies similar to one you liked based on genre or director.
- Recommend actors for a movie based on their past roles (using reasoning about relationships).

RDF (Resource Description Framework)

RDF is a standard model for data interchange on the web. It is used to represent information about resources in a graph-based format. RDF provides a way to describe relationships between resources in a subject-predicate-object form, allowing for the creation of semantic web applications that can understand and process this structured data.
Advantages of RDF:

- Flexibility: RDF's graph-based structure allows for flexible data modeling and representation.
- Interoperability: It promotes data interchange and interoperability between different systems and applications.
- Standardization: RDF provides a standardized way to represent data, enhancing consistency and compatibility.
- Semantic Web: RDF supports the development of semantic web applications that can process and understand structured data effectively.
- Scalability: RDF is scalable and can handle large datasets efficiently, making it suitable for diverse data-intensive applications

RDFS (Resource Description Framework Schema):

RDFS is an extension of RDF that provides a basic vocabulary for describing ontologies. It allows for defining classes, properties, and relationships between resources in a more structured manner.

Purpose of Classes in RDFS:

- rdfs:Resource: The class rdfs:Resource is the most general class in RDFS, representing any resource in the RDF model. It serves as the superclass for all resources in an RDF graph.
- rdfs:Class: The class rdfs:Class is used to define classes in RDFS, allowing for the categorization and classification of resources into different groups or types.
- rdfs:Domain: The property rdfs:Domain is used to specify the class that a property belongs to, indicating the domain of the property and defining the types of resources to which the property can be applied.
- rdfs:IsDefinedBy: The property rdfs:IsDefinedBy is used to link a resource to its defining ontology or schema, providing a reference to the location where the resource is formally defined.
- rdfs:comment: The property rdfs:comment is used to add human-readable comments or descriptions to resources, classes, or properties in an ontology, enhancing the understanding and documentation of the ontology's components.

Web Ontology Language (OWL):

OWL is a language used to define ontologies on the web. It allows users to create classes, properties, and relationships between entities. OWL is based on formal logic and is designed to be interpreted by machines, enabling automated reasoning and inference capabilities in applications that utilise ontologies.

Advantages of OWL:

- Formal Representation: OWL provides a formal and logical representation of knowledge, enabling precise definitions of concepts and relationships.
- Automated Reasoning: It supports automated reasoning processes, allowing for logical deductions and inferences. Semantic Web Development: OWL contributes to the development of the semantic web by providing a standardised language for defining ontologies.
- Knowledge Sharing: OWL facilitates knowledge sharing by defining explicit semantics that can be understood by both humans and machines.
- Interoperability: OWL promotes interoperability by providing a common framework for defining ontologies, enhancing data integration and sharing capabilities.

Data Science Framework

- The definition of the Data Science Framework involves a structured approach to converting raw unstructured data from data lakes into actionable business data. It is a cyclical process that focuses on discovering and evolving an understanding of the data to provide necessary metadata.
- The framework serves as a guide for constructing data science solutions and seamlessly transferring them to data engineering environments.
- It consists of layers that facilitate a logical building process, enabling the utilisation of data processing and discoveries across various projects.
- This framework is adaptable for projects of different scales, from small departmental initiatives to large international deployments, ensuring a systematic and effective approach to data science projects

Layered Framework

- The layered framework in the context of data science refers to a structured approach that involves different layers to enhance the quality and efficiency of data science projects.
- This framework consists of several layers, each serving a specific purpose in the data processing and analysis pipeline.
- The layers typically include the Business Layer, Utility Layer, Operational Management Layer, Audit, Balance, and Control Layer, and Functional Layer. Each layer plays a crucial role in ensuring that data science projects are well-organized, scalable, and aligned with business requirements.
- The layered framework provides a systematic guide for data scientists and engineers to follow, enabling them to build robust data solutions that can be seamlessly transferred to data engineering environments

Business Layer

- The business layer in data science serves as a crucial bridge between non-technical business requirements and the practical implementation of data science solutions.
- It involves converting business needs and objectives into specific data science requirements that can be addressed through analytical processes and technologies.
- The business layer is where the interactions with the business are recorded, translating high-level business goals into actionable data science tasks.
- **Functional Requirements**
    - Functional requirements within the business layer detail the specific criteria that need to be met to achieve the desired outcomes.
    - These requirements outline the functionalities, features, and capabilities that the data science solution must possess to address the identified business needs effectively.
    - They serve as a roadmap for developing the technical aspects of the solution and guide the implementation of data processing and analysis tasks.
- **Nonfunctional Requirements**
    - In addition to functional requirements, nonfunctional requirements are also essential components of the business layer.
    - Nonfunctional requirements define the quality attributes, constraints, and performance expectations of the data science solution.

    - These requirements focus on aspects such as scalability, reliability, security, usability, and performance efficiency.
    - Nonfunctional requirements ensure that the data science solution meets the necessary standards beyond just its functional capabilities.

Engineering a Practical Business Layer:

To engineer a practical business layer effectively, it is essential to follow general business analysis and project management principles. A well-structured business layer typically consists of three primary components:

- Requirements: Clearly defining and documenting both functional and nonfunctional requirements is crucial for building a successful data science solution. These requirements serve as the foundation for designing and implementing the technical aspects of the project.
- Requirements Registry: Establishing a requirements registry helps in organising and managing all identified requirements throughout the project lifecycle. This registry acts as a central repository for tracking, updating, and referencing requirements as needed.
- Traceability Matrix: Developing a traceability matrix enables linking each requirement back to its source in the business objectives or user needs. This matrix ensures that every aspect of the data science solution is aligned with the initial business requirements and goals.

Utility Layer

- The utility layer in data science serves as a repository for storing reusable and practical methods that facilitate common data processing tasks across various projects.
- It plays a crucial role in streamlining data processing methodologies, ensuring consistency, efficiency, and reliability in data science workflows. Basic Utility Design
- The utility layer is designed to house a collection of utilities that serve as the workhorses of the data science ecosystem.
- These utilities are essential tools that enable data scientists to perform repeatable tasks efficiently and effectively.
- By centralizing all utilities in one location, the utility layer ensures that outdated or duplicate algorithms are not used, promoting the use of stable and verified solutions across projects.

Types of Utilities Data

- Processing Utilities: These utilities focus on transforming data within solutions, handling tasks such as retrieving data from raw sources and structuring it for further analysis.
- Maintenance Utilities: Maintenance utilities are responsible for managing and maintaining the data processing ecosystem, ensuring smooth operations and system integrity.
- Processing Utilities: Processing utilities encompass tools that execute specific data processing tasks, enhancing the overall functionality and performance of the data science environment.

Engineering a Practical Utility Layer

To effectively engineer a utility layer, it is essential to follow best practices in organising and managing utilities:

- Centralised Storage: Collect all utilities, including source code, in a central repository to ensure easy access and version control.

- Version Control: Maintain detailed records of each utility version to track changes, updates, and ensure reliability.
- Quality Assurance: Keep proof and credentials demonstrating the quality and industry acceptance of algorithms used in utilities to validate their effectiveness.
- Standardization: Establish clear standards for utilities to enable multiple teams to work on parallel projects with consistent practices.

Operational Layer

The operational management layer in data science is a fundamental component that focuses on designing, controlling, and optimising the process chains within a production environment.
This layer plays a crucial role in orchestrating data processing pipelines, managing resources, and ensuring the efficient execution of data science workflows.
Key Functions of the Operational Management Layer:

- Processing-Stream Definition and Management: This function involves defining and managing the flow of data processing tasks within the data science ecosystem. It includes structuring the sequence of operations, managing dependencies, and optimising the processing stream for efficiency.
- Parameters: Managing parameters within the operational management layer involves configuring and controlling variables that influence data processing tasks. Parameters can include settings, thresholds, and constraints that impact the behaviour and outcomes of processing operations.
- Scheduling: Scheduling tasks and processes is a critical aspect of the operational management layer. It involves planning the execution of data processing activities, allocating resources, and ensuring that tasks are completed within specified timeframes to meet business requirements.
- Monitoring: Monitoring the performance and status of data processing operations is essential for maintaining system health and efficiency. The operational management layer includes mechanisms for real-time monitoring, tracking progress, identifying bottlenecks, and ensuring that processes are running as expected.
- Communication: Facilitating communication within the data science environment is another key function of the operational management layer. Effective communication channels enable collaboration, coordination, and information sharing among team members involved in data processing tasks.
- Alerting: Alerting mechanisms within the operational management layer provide notifications and warnings about critical events, anomalies, or issues that require immediate attention. Alerts help data scientists and engineers proactively address problems and maintain the integrity of data processing operations

Best Practices for Operational Management:

- Efficient Resource Allocation: Optimise resource allocation to ensure that processing tasks are executed with maximum efficiency and minimal resource wastage.
- Real-Time Monitoring: Implement real-time monitoring tools to track the performance of data processing operations, identify issues promptly, and take corrective actions.
- Scalability Planning: Design processing pipelines with scalability in mind to accommodate varying workloads and ensure that the system can handle increased demands effectively.

- Automation: Utilise automation tools and scripts to streamline repetitive tasks, reduce manual intervention, and improve the overall efficiency of data processing workflows.

Audit, Balance, and Control Layer

The audit, balance, and control layer is a critical component within the data science ecosystem that focuses on monitoring, managing, and ensuring the integrity and efficiency of data processing operations. This layer plays a pivotal role in maintaining transparency, accountability, and compliance within data science projects, enabling data scientists and engineers to track, analyse, and optimise the performance of their processes effectively.

Audit Sublayer:

The audit sublayer within the ABC layer is responsible for recording and tracking all processes executed within the data science environment. It serves as a valuable resource for data scientists and engineers to understand the flow of operations, identify potential bottlenecks, and plan improvements to enhance processing efficiency. A robust audit trail is essential in providing a comprehensive record of activities, facilitating troubleshooting, and ensuring accountability in data processing tasks.

Balance Sublayer:

The balance sublayer ensures that the data science ecosystem remains balanced in terms of processing capabilities and resources. It focuses on maintaining equilibrium across available processing resources, ensuring that the system can handle varying workloads efficiently. By leveraging capabilities such as processing-on-demand in cloud environments, the balance sublayer enables dynamic resource allocation to meet changing processing requirements effectively.

Control Sublayer:

The control sublayer is dedicated to implementing mechanisms that govern system access, manage errors, and ensure compliance with regulations and privacy standards. Role-based access control (RBAC) is a commonly used approach within the control sublayer, restricting system access to authorised users based on predefined roles and privileges. Additionally, control mechanisms encompass error handling, code management, and adherence to privacy laws and regulations to safeguard data integrity and security.

Best Practices:

- Maintaining a Comprehensive Audit Trail: Ensuring that all processes are logged and documented to provide a clear record of activities for analysis and improvement.
- Dynamic Resource Allocation: Implementing strategies for balancing processing capabilities to meet fluctuating workloads efficiently.
- Adherence to Privacy Regulations: Following strict privacy laws and regulations to protect sensitive data and ensure compliance with legal requirements.
- Role-Based Access Control: Establishing role-based access control mechanisms to manage system access and permissions effectively.

Functional Layer

The functional layer within the data science ecosystem is a pivotal component responsible for executing comprehensive data science processes.

This layer encompasses various structures, including data models, processing algorithms, and infrastructure provisioning, essential for conducting data analysis and deriving actionable insights. Key Components of the Functional Layer:

- Data Models: Data models within the functional layer define the structure and relationships of the data being analysed. These models help organise and represent data in a way that facilitates analysis and interpretation, enabling data scientists to derive meaningful insights from complex datasets.
- Processing Algorithms: Processing algorithms form a crucial part of the functional layer, encompassing the computational methods and techniques used to manipulate and analyse data. These algorithms are designed to perform tasks such as data transformation, pattern recognition, and predictive modelling, enabling data scientists to extract valuable information from raw data.
- Infrastructure Provisioning: Infrastructure provisioning in the functional layer involves setting up the necessary computing resources, software tools, and environments required to support data processing tasks. This includes configuring data storage systems, computing clusters, and software frameworks to facilitate efficient data analysis and processing.

Supersteps of Processing in the Functional Layer:
The processing algorithm in the functional layer is typically organised into distinct supersteps, each serving a specific purpose in the data analysis workflow:

- Retrieve: This superstep involves retrieving data from raw data sources, transforming it into a structured format for further analysis.
- Assess: The assess superstep focuses on quality assurance and enhancing data quality through additional processing steps.
- Process: Processing superstep involves building the data vault, a structured repository for storing processed data.
- Transform: Transform superstep focuses on transforming data into a format suitable for building the data warehouse.
- Organise: Organise superstep involves structuring data into data marts, specialised subsets of data for specific analytical purposes.
- Report: Report superstep focuses on generating actionable insights and visualisations from the processed data, enabling stakeholders to make informed decisions based on the analysis.

DBR focuses on synchronizing production processes by identifying and managing the constraint (the "drum") in the system. Here's how it works in six phases:

1. Identify the Constraint: The first and crucial step is pinpointing the bottleneck or limitation in the production process. This could be a machine, a process step, or material availability.

2. Set the Drumbeat: Based on the constraint's capacity, set a production schedule (the "drumbeat") that dictates the overall pace of production.

3. Create Buffers: Establish safety stock (buffers) of materials before and after the constraint to ensure it never runs out of work.

4. Release Kanban Cards: Implement Kanban cards or a similar system to authorize the release of new work into the system. Only enough work is released to maintain a smooth flow based on the drumbeat and buffer levels.

5. Continuous Improvement: Monitor and continuously improve the process. Analyze data to identify opportunities for reducing lead times, eliminating waste, and potentially increasing the capacity of the constraint.

6. Replenish Buffers: Regularly replenish buffer stocks to maintain their target levels.

UNIT 4

Sums

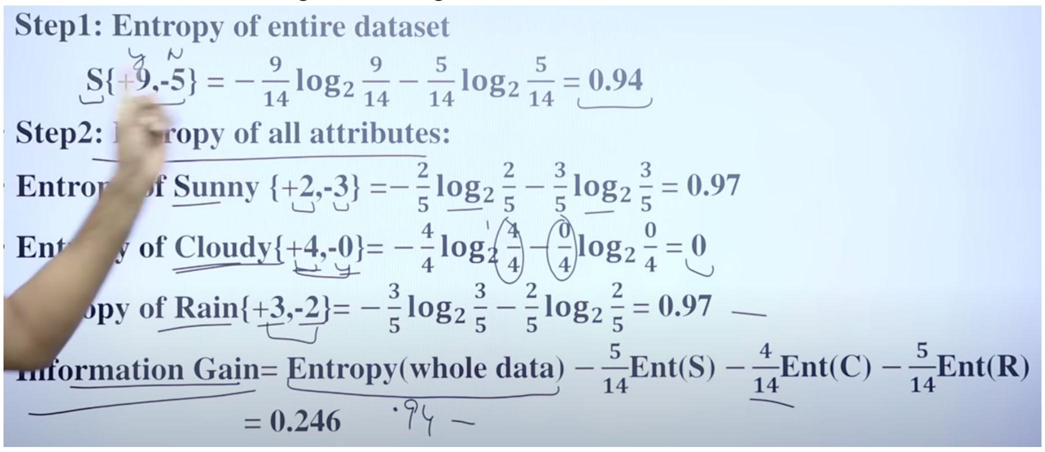
Linear Regression

Main formula = Y = Mx +b
b = Mean(y) - (M * Mean(x))
M = Sum of multiplied values of deviations / Sum of squared values of deviation (x)
[Basically second last column / last column]
First get data and make this table

x	y	Mean(x)	Mean(y)	Deviation(x)from mean	Deviation(y)from mean	Multiply both deviations x and y	Sum of multiplied values	Sum of squared deviation of (x)

ID3

Calculate entropy and info gain for each thi leads to making of Descion tree
Highest info gain is the answer == root node of tree
Entropy of full dataset
In sum lets say 9 yes and 5 nos
Entropy == S = **- 9/14 * Log$_2$ 9/14 - 5/14 Log$_2$ 5/14** = 0.94
If one column is weather with three rainy , Cloudy ,Sunny
For each calculate personal Entropy with above formula
Information gain (weather)= Entropy(whole data set(S)) -(Total rows of sunny /Total Rows in weather)Entropy(Sunny) -(Total rows of cloudy /Total Rows in weather)Entropy(Cloudy) - (Total rows of Rainy /Total Rows in weather)Entropy(Rainy)
Do for all Select the Highest info gain



Step1: Entropy of entire dataset

$\underbrace{S\{+\overset{y}{9},-\overset{N}{5}\}} = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = \underbrace{0.94}$

Step2: [illegible]ropy of all attributes:

Entro[illegible]f Sunny $\{+2,-3\} = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.97$

Ent[illegible]y of Cloudy$\{+4,-0\} = -\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4} = 0$

[illegible]opy of Rain$\{+3,-2\} = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0.97$ —

Information Gain= Entropy(whole data) $- \frac{5}{14}$Ent(S) $- \frac{4}{14}$Ent(C) $- \frac{5}{14}$Ent(R)

= 0.246 .94 –

Types of regressions

Linear Regressions

● Linear regression is a data analysis technique that predicts the value of unknown data by using another related and known data value.
● It mathematically models the unknown or dependent variable and the known or independent variable as a linear equation.
● For example, suppose that you have data about your expenses and income for last year. Linear regression techniques analyze this data and determine that your expenses are half your income. They then calculate an unknown future expense by halving a future known income.

Polynomial Regressions

- Polynomial regression is a technique in statistics and machine learning used to model non-linear relationships between an independent variable (x) and a dependent variable (y). It achieves this by fitting a polynomial equation to the data, allowing for curves and bends rather than just a straight line like linear regression.
- Examples :
  - - Predicting growth trends (e.g. population growth over time)
  - - Analyzing economic trends with nonlinear patterns (e.g. Supply and demand)

Logistic Regressions

- Data scientists use logistic regression to measure the probability of an event occurring. The prediction is a value between 0 and 1, where 0 indicates an event that is unlikely to happen, and 1 indicates a maximum likelihood that it will happen. Logistic equations use logarithmic functions to compute the regression line.
- Examples:
  - The probability of a win or loss in a sporting match
  - The probability of passing or failing a test
  - The probability of an image being a fruit or an animal

Sr. No	**Linear Regression**	**Polynomial Regression**	**Logistic Regression**
1	Used for predicting relationships between variables	Used for capturing non-linear relationships	Used for classification of data into categories
2	Linear type of function	Non- Linear type of function	Sigmoidal type of function
3	Works well when relationship between variables is linear	Can capture more complex relationships	Suitable for binary or multi-class classification problems
4	Simple to understand and implement	More complex than linear regression	Slightly more complex than linear regression
5	Prone to overfitting with complex data	Prone to overfitting with high degree polynomials	Regularisation techniques often applied to prevent overfitting

Confusion Matrix
A confusion matrix, also sometimes called an error matrix, is a table that provides a visual and insightful breakdown of a classification model's performance. It's particularly useful in supervised learning tasks where the model is trained to categorise data points into predefined classes.
Key Elements:

- True Positives (TP): These are data points that the model correctly classified as belonging to the positive class.
- True Negatives (TN): These are data points that the model correctly classified as belonging to the negative class (or any other non-positive class if it's a multi-class problem).
- False Positives (FP): These are data points that the model incorrectly classified as positive when they actually belong to the negative class (or another non-positive class). These are also known as **Type I errors**.
- False Negatives (FN): These are data points that the model incorrectly classified as negative when they actually belong to the positive class. These are also known as **Type II errors**.

Benefits:

- Visualization: Provides a clear visual representation of the model's classification performance.
- Detailed Insights: Goes beyond simple accuracy to reveal class-specific strengths and weaknesses.
- Error Analysis: Helps identify areas for improvement in the model.

Accuracy = (TP + TN) / (Total Samples)
Error Rate = 1 - Accuracy
Precision = TP / (TP + FP) [tp + total retrieved]
Recall = TP / (TP + FN)[tp +total relevant]
F1 measure = 2PR /P+R

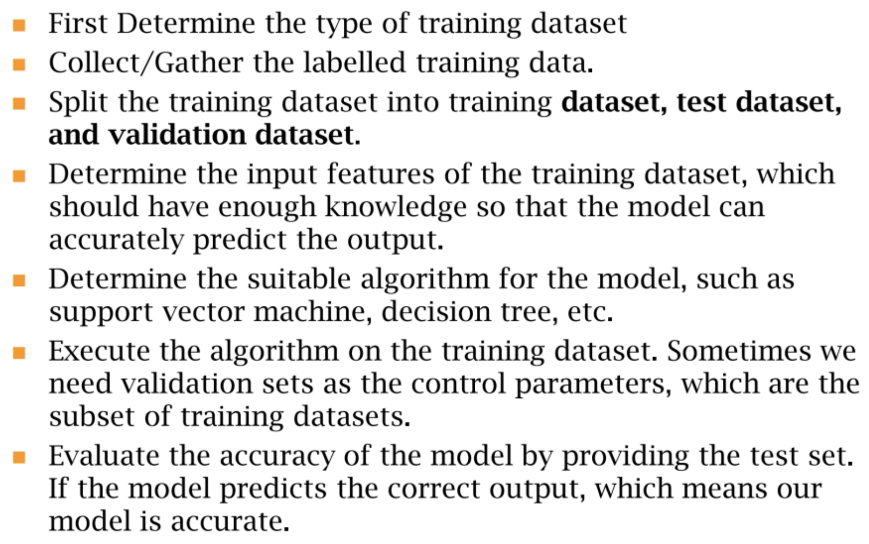
Training and Testing Data

- Training Data: A portion of the dataset used to train the machine learning model. The model learns patterns and relationships from this data.
- Testing Data: A separate portion of the dataset used to evaluate the model's performance on unseen data. This helps assess how well the model generalises to new examples.

Supervised and Unsupervised Learning

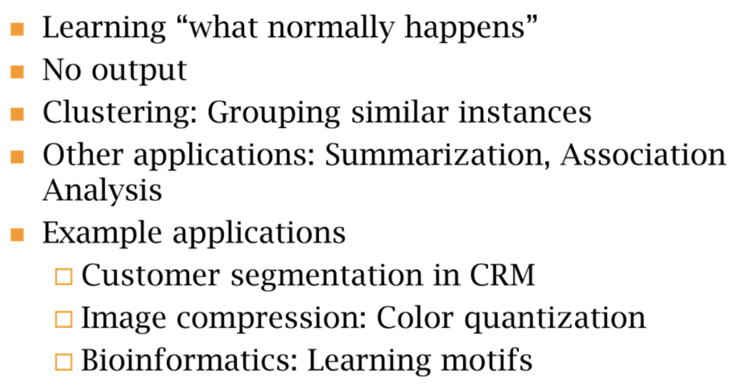
Supervised Learning: The model learns from labelled data, where each data point has a corresponding label (target variable). The model learns to map inputs to desired outputs. (e.g., image classification, spam detection)

Steps involved in Supervised Learning :

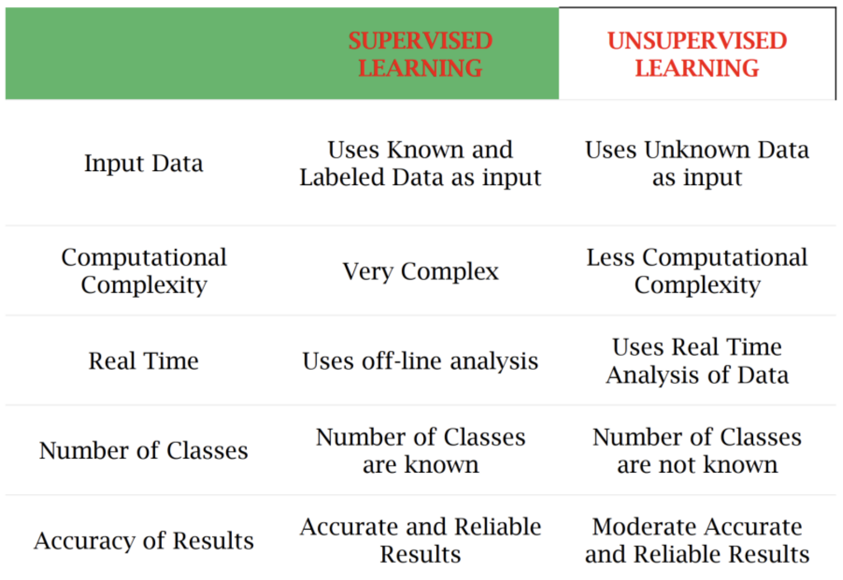


- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training **dataset, test dataset, and validation dataset.**
- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset. Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set. If the model predicts the correct output, which means our model is accurate.

Unsupervised Learning:
The model deals with unlabeled data, where the data
points lack predefined labels. The model aims to discover hidden patterns or structures within the data. (e.g., anomaly detection, customer segmentation)



- Learning "what normally happens"
- No output
- Clustering: Grouping similar instances
- Other applications: Summarization, Association Analysis
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs

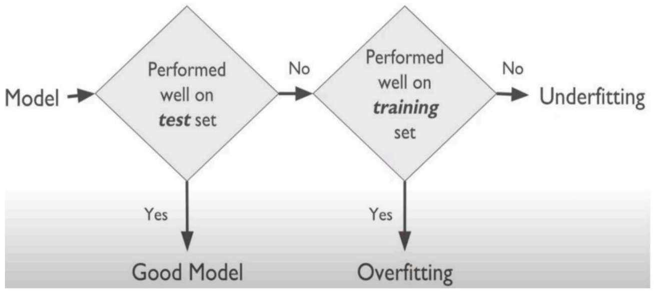


	SUPERVISED LEARNING	UNSUPERVISED LEARNING
Input Data	Uses Known and Labeled Data as input	Uses Unknown Data as input
Computational Complexity	Very Complex	Less Computational Complexity
Real Time	Uses off-line analysis	Uses Real Time Analysis of Data
Number of Classes	Number of Classes are known	Number of Classes are not known
Accuracy of Results	Accurate and Reliable Results	Moderate Accurate and Reliable Results

Overfitting, Underfitting, and Perfect Fit

- Overfitting: Occurs when a model memorizes the training data too closely, failing to capture the underlying relationships and performing poorly on new data. It's like studying only past exam questions without understanding the concepts.

- Underfitting: Occurs when a model is too simple to capture the true relationship between the features and target variable, resulting in high errors on both training and testing data. It's like not studying enough for the exam.
- Perfect Fit: (Not recommended) While theoretically possible, a model that perfectly fits the training data will always overfit and perform poorly on unseen data. It's like memorizing every single past exam question word-for-word, which won't help if the questions change slightly.



Bias and Variance

- Bias: The tendency of a model to consistently under- or over-predict the target variable. It's like having a systematic error in your approach. High Bias: The model is too simple and misses the real relationship (underfitting).
- Variance: The tendency of a model to be sensitive to the specific training data, leading to high variability in its predictions on different training sets. It's like your results depending heavily on the specific questions you studied. High Variance: The model is too complex and memorizes noise in the data (overfitting).

Binary Classification:

- Main Goal: Classifies data points into exactly two categories. These categories are often denoted as positive (1) and negative (0), or any other two distinct labels.
- Applications: Spam filtering (spam or not spam), image recognition (cat or dog), sentiment analysis (positive or negative review).
- Examples of Algorithms: Logistic Regression, Support Vector Machines (SVM), Decision Trees.

Lazy Learning (Instance-Based Learning):

- Execution Style: Delays the model building process until a new data point needs classification.
- Mechanism: Relies on memorizing the training data and comparing new data points to the stored examples during prediction.
- Common Algorithm: K-Nearest Neighbors (KNN) - classifies a new point based on the majority vote of its K nearest neighbors in the training data.
- Advantages: Simple to implement, can handle complex decision boundaries, works well with high-dimensional data.
- Disadvantages: Can be computationally expensive for prediction due to distance calculations, requires storing all training data.

Eager Learning (Model-Based Learning):

- Execution Style: Builds a model from the training data upfront before receiving any new data for prediction.
- Mechanism: Learns a general representation of the data and applies this model to classify new data points.
- Common Algorithms: Decision Trees, Logistic Regression, Neural Networks.
- Advantages: Generally faster for prediction compared to lazy learning, can provide insights into the data through the model itself.

- Disadvantages: May struggle with complex data or require careful model selection and tuning to avoid overfitting.

Linear Regression vs. Non-Linear Regression:

Linear Regression:

- Assumes a straight-line relationship between independent (predictor) and dependent (response) variables.
- Simple to interpret, but limited to modeling linear relationships.

Non-Linear Regression:

- Captures more complex relationships between variables that may not be linear.
- Offers greater flexibility, but can be more challenging to interpret and prone to overfitting.

Simple Linear Regression vs. Multiple Linear Regression:

Simple Linear Regression:

- Models the relationship between just one independent variable and the dependent variable.
- Easier to understand and visualize.

Multiple Linear Regression:

- Considers the impact of multiple independent variables on the dependent variable.
- Provides a more comprehensive understanding of influencing factors.

Linear Regression vs. Logistic Regression:

Linear Regression:

- Predicts continuous numerical values for the dependent variable.
- Often used for tasks like forecasting sales or predicting house prices.

Logistic Regression:

- Predicts the probability of an event occurring, typically with a binary outcome (e.g., yes/no, pass/fail).
- Commonly used for classification problems like fraud detection or email spam filtering.