BAOU Education for All

Dr. Babasaheb Ambedkar Open University

(Established by Government of Gujarat)

NAAC ACCREDITED A++ GRADE 3.55/4

FUNDAMENTALS OF DATA SCIENCE

BCAMI-404

- DATA COLLECTION
- DATA PROCESSING
- ANALYSIS
- INSIGHTS
- APPLICATIONS

Bachelor of Computer Application (BCA)

Course Writer

Dr. Sanskruti Patel Professor and I/C Dean, Smt. Chandaben Mohanbhai Patel Institute of Computer Applications (CMPICA), Faculty of Computer Science and Applications (FCA), Charotar University of Science and Technology, Changa, Gujarat
Dr. Dharmendra Patel Professor and I/C Principal, Smt. Chandaben Mohanbhai Patel Institute of Computer Applications (CMPICA), Faculty of Computer Science and Applications (FCA), Charotar University of Science and Technology, Changa, Gujarat

Content Editor & Reviewer

Prof. (Dr.) Nilesh Modi Professor and Director, School of Computer Science, Dr. Babasaheb Ambedkar Open University, Ahmedabad

Expert Committee

Prof. (Dr.) Nilesh Modi Professor and Director, School of Computer Science, Dr. Babasaheb Ambedkar Open University, Ahmedabad	**(Chairman)**
Prof. (Dr.) Ajay Parikh Professor and Head, Department of Computer Science Gujarat Vidyapith, Ahmedabad	**(Member)**
Prof. (Dr.) Satyen Parikh Dean, School of Computer Science and Application Ganpat University, Kherva, Mahesana	**(Member)**
Prof. M. T. Savaliya Associate Professor and Head, Computer Engineering Department Vishwakarma Engineering College, Ahmedabad	**(Member)**
Dr. Himanshu Patel Assistant Professor, School of Computer Science, Dr. Babasaheb Ambedkar Open University, Ahmedabad	**(Member Secretary)**

ISBN- 978-81-986947-5-1

Printed and published by: Dr. Babasaheb Ambedkar Open University, Ahmedabad

Dr. Babasaheb
Ambedkar
Open University

BCAMI-404

Fundamentals of Data Science

Block–1

Block–2

Block–3

Block–4

Block-1

Introduction to Data Science

Unit-1: Overview of Data Science

Unit Structure

1.0 Learning Objectives

After studying this unit student should be able to:

- Understand the concept of Data Science and explain its role in modern computing and decision-making.
- Identify the key components of Data Science, including data collection, data cleaning, analysis, and visualization.
- Aware about Skillsets required for Data Science
- Able to aware about general and well-known issues of various types of Data.

1.1 Introduction

Data is all around us in today's digital world, from the messages we send and the films we watch to online banking, shopping, and even medical information. The area of data science is what enables us to easily and meaningfully make sense of this enormous volume of information. It entails gathering information, recognizing trends, and applying those trends to provide answers or assist in making better choices. For instance, information can help a company comprehend consumer preferences, help physicians see health trends, or let apps suggest movies and goods we would find interesting. Crucially, data science is an interdisciplinary profession that integrates logical reasoning, fundamental mathematics, and domain expertise from industries like business, healthcare, and social sciences; it is not just for programmers or technical specialists. A number of Data Science definitions exist in the literature. Here, some important definitions are mentioned.

Definition 1: Data Science is the process to better understand problems using data to and make efficient and decisions.

Definition 2: Data Science is the study of gathering, evaluating, and interpreting data in order to derive valuable insights.

Definition 3: Data Science helps organizations use data to make wise and informed decisions.

Definition 4: Data science forecasts future events using data, statistics, and algorithms.

Definition 4: Data science forecasts future events using data, statistics, and algorithms.

Definition 5: Data science combines programming, statistics, and subject-matter expertise to extract meaningful data.

Definition 6: Data Science is the study of data using scientific techniques, procedures, and tools to address practical issues.

Definition 7: Data Science is all about learning from data to understand the past, analyze the present, and predict the future.

The first definition suggests that instead of depending just on guessing or assumptions, we use data to investigate a situation carefully. Numerous sources, including surveys, websites, mobile apps, company records, and even everyday behaviors, might provide data. We are able to clearly see patterns, trends, and linkages by organizing and analysing this data. For instance, a store owner might be curious in the best-selling items. They can examine historical sales data rather than speculating. This aids them in determining which products to stock and in understanding consumer preferences. Similar to how businesses can use consumer data to enhance services, hospitals can utilize patient data to enhance treatments. “Better understanding difficulties" refers to dissecting an issue and using data to research it. Making precise, fast, and advantageous decisions while lowering risks and errors is what is meant by "efficient decisions. “To put it simply, data science enables us to transform unprocessed data into meaningful information that helps us make better decisions in both business and daily life.

The second definition of data science is working with data step-by-step in order to comprehend and make use of it.To begin with, data collection entails acquiring information from many sources. Online purchasing websites, social media, surveys, school records, and mobile apps are some of the sources of this data. For instance, a business might gather information about the goods that consumers purchase. Second, data evaluation entails accurate data organization and verification. Errors, missing values, or extraneous information can occasionally be found in data. This stage involves cleaning and organizing the data to make it easier to use and comprehend. Third, evaluating data entails examining the data to determine its significance. We search for relationships, patterns, or trends. For instance, we might discover that consumers purchase more cold beverages during the summer than during the winter. Ultimately, drawing meaningful inferences from the data is what it means to get important insights. Making smarter decisions is aided by these insights. For example, depending on the data, a store owner might decide to stock more cool drinks in the summer. To put it simply, data science is the process of gathering, cleaning, analysing, and applying data to generate practical concepts or solutions.

According to the third description, businesses, hospitals, educational institutions, and other organizations can make judgments without relying solely on experience or conjecture.

Alternatively, they can be guided by data. Utilizing data entails gathering information regarding various actions. For instance, a hospital would gather medical records, a business might gather information about consumer purchases, and a school might gather information on student performance. Decisions that are based on facts rather than conjecture, more accurate and trustworthy, and helpful in lowering errors and dangers are referred to as wise and informed decisions. A shopping website, for instance, can examine consumer data to determine what consumers prefer to purchase. They can provide product recommendations to clients based on this. This boosts client happiness and helps the business generate sales. In a similar vein, a college can evaluate student performance to enhance teaching strategies, and a hospital might examine patient data to determine more effective treatments. To put it simply, data science enables businesses to comprehend their data and make better, more informed decisions.

According to the fourth definition, data science uses historical and current data to forecast potential future events. First, data is the information we gather from many sources, including user activity, weather reports, and sales records. What has previously occurred is shown by this data. Second, data is summarized and comprehended using statistics. It assists us in identifying trends, averages, and patterns. For instance, we can observe whether sales are rising each month or whether some products are more well-liked. Third, algorithms are computer programs or methodical approaches that evaluate the data and generate predictions. These algorithms forecast future results by learning from historical data. Therefore, Data Science use appropriate techniques to create logical and data-based forecasts rather than speculating.

The fifth definition states that while statistics aids in the analysis of data to identify patterns, trends, and relationships, data science uses computer programs (programming) to handle and process massive amounts of data. Subject-matter expertise, or understanding of a particular field, such as business, healthcare, or education, is crucial to comprehending the meaning of the data and its practical applications. Unorganized and raw data can be transformed into valuable knowledge when these three domains collaborate. This facilitates problem-solving and improved decision-making. For example, a corporation might analyse customer data to boost sales, or a hospital can examine patient data to give better treatment. In summary, Data Science is about using the correct tools and knowledge to turn data into important insights.
The sixth definition is that data science is a methodical approach to using data to address practical issues. It adopts a methodical approach (procedures) and employs appropriate methods (scientific techniques) in addition to technologies like computers and software to

thoroughly examine data rather than making haphazard guesses. These techniques help us comprehend issues more fully and come up with practical solutions. For instance, a hospital can use data to improve patient care, a business can use it to increase sales, and a college can use student performance analysis to improve instruction. To put it briefly, data science enables us to efficiently solve common problems by using data in an organized and rational manner.

As per the seventh Definition, Data Science facilitates the use of data as a learning resource. We can comprehend what has previously occurred by looking at historical data. We can comprehend the current situation by examining the data. Additionally, we can forecast potential future events by analysing patterns and trends in this data. For instance, a business can examine historical sales data to gain insight into consumer behaviour, examine current patterns to identify popular products, and then forecast future demand. Similar to this, data is used in business planning, recommendation systems, and weather forecasting to improve decision-making. To put it briefly, data science enables us to learn from data at every level—past, present, and future—in order to make more intelligent decisions.

Summary of all above definitions is depicted in the table 1.1.1

Table 1.1.1 Summary of Data Science Definitions

Definition	**Key Idea**	**Meaning**	**Focused Area**
1	Understanding problems & decision-making	Using data to understand problems and make better and efficient decisions	Problem-solving
2	Data collection & analysis	Collecting, checking, and understanding data to get useful insights	Data handling
3	Informed decisions	Helping organizations make smart decisions based on data	Decision-making

4	Prediction	Using data, statistics, and algorithms to predict future outcomes	Forecasting
5	Combination of skills	Using programming, statistics, and domain knowledge to understand data	Multidisciplinary
6	Scientific approach	Using proper methods, tools, and steps to solve real-world problems with data	Systematic process
7	Past–Present–Future analysis	Learning from past data, analysing current data, and predicting future trends	Complete data lifecycle

1.2 Evolution of Data Science

Data Science did not emerge suddenly; it has evolved over time through the development of statistics, computing, and data technologies. The journey shows how humans moved from simple data recording to advanced intelligent systems.

The first evolution was known as Early Data Collection. This era was before 1960s. All data was gathered and kept manually in the early phases of data management, prior to the widespread usage of computers. Physical formats including registers, notebooks, folders, and ledgers were used by people to record information. Businesses kept accounts in handwritten books, governments gathered population statistics through census operations, and organizations used paper-based systems to keep track of transactions, personnel, and inventory. Because even minor errors could result in inaccurate records, this procedure required a great deal of human labour, time, and careful attention. Record-keeping, rather than analysis, was the primary goal of data collecting throughout this time. The majority of decisions were based on basic calculations and observations because there were relatively few instruments available

for data processing or analysis. Because physical records needed a lot of storage space and were hard to arrange and retrieve, data storage was also a problem. It could take a while to find precise information, and there was always a chance that data would be lost because of fire, damage, or misplacement. Despite these drawbacks, the groundwork for contemporary data science was established by early data collection. It emphasized the need for improved tools and techniques to manage and analyse data and assisted companies in realizing the need of keeping correct records. The transition from basic data recording to the sophisticated, technologically advanced data analysis of today starts at this point.

Key Features

- Data recorded in registers and files
- Basic calculations done manually
- Limited data storage capacity

Example

- Census data collection
- Business accounting records

The second evolution was known as Development of Statistics. The era of 1960s to 1980s is known as second evolution of Data Science. Statistics emerged as the primary instrument for comprehending and evaluating data between the 1960s and the 1980s. This stage concentrated on utilizing mathematical techniques to make meaning of data, as opposed to previous periods when data was just recorded. To summarize data and spot trends, researchers, corporations, and governments started using statistical methods including mean, median, standard deviation, regression, and correlation. This made it easier to transform unprocessed data into insightful knowledge that may assist make better decisions. The invention of computers for data processing was one of the most significant advancements during this time. Calculations could be completed far more quickly with early computers than with manual methods, despite their size, cost, and power limitations. This made it possible for analysts to work with bigger datasets and decreased human effort and error. As a result, statistical analysis grew more useful and common in a variety of disciplines, including social sciences, education, healthcare, and economics. The creation of statistical software tools like SAS and SPSS was another significant component of this phase. With the help of these tools, users may more easily use statistical techniques without having to conduct intricate computations by hand. Data entry, analysis, and report generation might all be done more effectively by users. As a result, statistics became more accessible, enabling even non-experts to start working with data.

Key Features

- Use of mathematical formulas
- Data summarized using averages, graphs, and tables
- Early computers introduced

Tools

- Statistical methods (mean, variance, regression)
- Software like SPSS and SAS

The third evolution was considered from 1980s to 1990s where Databases introduced. Better methods for managing and storing data were required during the 1980s and 1990s due to the quick expansion of computers and commercial activities. In the past, information was frequently kept in distinct files, making it challenging to arrange, update, and retrieve data. As a result, Database Management Systems (DBMS) were created, making it possible to store data in an orderly and structured way. Data could now be stored in tables with rows and columns rather than dispersed files, which made management simpler. The development of Relational Database Management Systems (RDBMS) was one of the most significant developments during this time. Data is kept in several linked tables in this system, and keys—such as primary and foreign keys—are used to establish associations. Reducing data duplication and preserving data consistency were made achievable by this structure. During this period, well-known database systems like Oracle, MySQL, and Microsoft SQL Server appeared and spread throughout businesses.

Key Features

- Introduction of Relational Database Management Systems (RDBMS)
- Structured data storage
- Use of SQL for querying data

Examples

- Banking systems
- Inventory management

The Data Mining era during 1990s to 2000s was known as fourth evolution of the Data Science. Organizations realized in the 1990s and early 2000s that merely storing data was insufficient; they also needed to extract valuable insights from it. As a result, the era of data mining emerged. Finding hidden patterns, connections, and trends in big databases is known as data mining. The development of databases and increased processing power made it feasible to examine vast

amounts of data and find insightful information that was previously difficult to see. Several data mining methods and algorithms were created during this time to improve data analysis. These included techniques like anomaly detection (finding odd patterns), association rule mining (finding associations between things), clustering (finding similar groups in data), and classification (grouping data into categories). For instance, retailers employed market basket analysis—a type of data mining—to determine which products are frequently bought together. This aided companies in enhancing customer satisfaction and sales tactics. The integration of data mining with business intelligence (BI) tools was another significant advancement. Organizations began presenting data analysis results in an easy-to-understand manner by employing dashboards, reports, and visualization tools. This made it possible for managers and decision-makers to forecast, plan, and enhance corporate operations using data insights. Many industries, including banking (fraud detection), marketing (client segmentation), healthcare (disease pattern analysis), and telecoms (user churn prediction), have made extensive use of data mining.

Key Features

- Discovery of patterns and relationships
- Use of algorithms for classification and clustering
- Growth of business intelligence

Applications

- Market basket analysis
- Fraud detection

The fifth evolution was considered during 2000s to 2010s where concept of Big Data was emerged. The volume of data increased rapidly due to the internet and digital platforms. Every activity began producing massive amounts of data every second, including social media interactions, banking transactions, internet purchases, and sensor data from devices. Big Data emerged as a result of traditional data processing systems' inability to handle such large and complicated data. New technologies and tools were developed. Systems like Hadoop and NoSQL databases allowed organizations to store and process large datasets efficiently. Unlike traditional databases, these systems could handle unstructured and semi-structured data, making them more flexible and scalable. The transition to distributed computing, which processes data over several machines rather than a single system, was another significant aspect of this phase. Real-time or nearly real-time analysis was made possible by this increased speed and efficiency.

Key Features

- Massive data from social media, sensors, and web
- Introduction of Big Data technologies
- Need for scalable storage and processing

Technologies

- Hadoop
- NoSQL databases

The sixth era was known as modern data science. The era from 2010 to present is known as modern data science. The most sophisticated phase of the development of dealing with data is represented by modern data science. These days, data is used to create intelligent systems, automate choices, and make forecasts in addition to being examined to understand historical trends. Massive volumes of data have been produced as a result of the quick development of digital technologies, internet usage, cloud computing, and smart gadgets. These data are currently being efficiently used with the help of contemporary tools and methods. The integration of several disciplines, including as programming, statistics, mathematics, and domain expertise, is one of the main characteristics of contemporary data science. While statistical techniques aid in the analysis of patterns and relationships, data scientists handle and process data using computer languages like Python and R. Simultaneously, expertise in a particular industry (like business, finance, or healthcare) aids in comprehending the issue and successfully implementing the findings. The application of artificial intelligence (AI) and machine learning (ML) is another significant advancement at this point. Without explicit programming, these technologies let systems to automatically learn from data and get better over time. Machine learning models, for instance, power voice assistants, fraud detection in banks, and recommendation systems on e-commerce websites. AI's deep learning component is employed for increasingly difficult tasks like natural language processing and picture identification. Large-scale and real-time data processing is another major emphasis of modern data science. Distributed systems and cloud platforms enable businesses to rapidly evaluate data as it is created. Applications like stock market analysis, internet transactions, and real-time monitoring systems benefit greatly from this. Widely used tools include advanced analytics platforms, cloud services (AWS, Azure, Google Cloud), and Apache Spark.

The Table 1.2.1 describes the summary of evolution of Data Science.

Table 1.2.1 Summary of Evolution of Data Science

Duration	**Stage**	**Focus**
Before 1960s	Manual	Record Keeping
1960-1980	Statistics	Data Analysis
1980-1990	Databases	Data Storage
1990-2000	Data Mining	Pattern Discovery
2000-2010	Big Data	Large Scale Data Handling
2010-Present	Modern Data Science	Prediction and Artificial Intelligence

Check Your Progress-1

a. Define any one Definition of Data Science

b. Data science forecasts future events using data, statistics, and algorithms (True/False).

c. Only Programming skill is important for Data Science(True/False)

d. _____ emerged as the primary instrument for comprehending and evaluating data between the 1960s and the 1980s

e. What are the key features of third evolution of Data Science?

f. Which technologies are used in Modern Data Science Evolution?

1.3 Importance of Data Science

Data Science play a vital role in various tasks of modern life. Data Science is crucial for various tasks.

- **More Informed Decisions:** People and organizations can make better, more informed decisions with the aid of data science. Decisions are founded on facts and data analysis rather than conjecture or intuition. For instance, businesses use consumer data to determine which items to introduce or cancel. This raises the likelihood of success while lowering hazards.
- **Understanding Customer for Businesses:** Understanding customers is one of the main benefits of data science for businesses. Businesses can determine the preferences and wants of their customers by examining data like past purchases, browsing habits, and reviews. This enables them to offer individualized services, raise client happiness, and create enduring bonds.

- **Forecasting Trends:** Data science makes it possible to forecast future results using historical data. Companies are able to predict market trends, demand, and sales. E-commerce businesses, for instance, are able to forecast which goods will be popular at a certain time of year. Planning and resource management are aided by this.
- **Encouraging Innovation and Development:** Innovation heavily relies on data science. It aids businesses in finding new prospects, creating new goods, and enhancing current offerings. Data science is the foundation of many contemporary technologies, including chatbots, recommendation systems, and smart products.
- **Increasing Productivity and Efficiency**: Data science can be used by organizations to enhance their operations. They can find inefficiencies, cut waste, and improve procedures by studying data. For instance, the manufacturing sector uses data to lower costs and enhance production procedures.
- **Managing Huge Data Sets:** A massive volume of data is produced every second in the current digital era. Data science offers methods and tools for effectively managing, processing, and analyzing this massive amount of data. Managing such data would be extremely challenging without data science.
- **Competitive Advantage for Organization:** Employing data science gives businesses a competitive advantage. Effective data analysis enables them to comprehend market trends, enhance plans, and make judgments more quickly than their rivals.
- **Intelligence Systems and Automation:** Through artificial intelligence and machine learning, data science facilitates automation. Systems are able to learn from data and carry out tasks automatically, including speech recognition, fraud detection, and product recommendations. This minimizes human work and saves time.
- **Improved Risk Control:** Data science facilitates the identification of possible hazards and the implementation of preventative measures. For instance, businesses employ data analysis to control financial risks, while banks use it to identify fraudulent transactions.

1.4 Key Components of Data Science

Data Science comprises of various components. First and important component is Data Collection. Gathering information from various sources, such as websites, apps, surveys, or records, is known as data collecting. This is where data science begins. Because better data produces better results, high-quality data is crucial. Methods of Data Collection are divided into two main categories: Qualitative and Quantitative. Qualitative Data Collection methods

involve face to face personal interview, Qualitative Surveys, Documental Revision, Case Studies etc. Quantitative techniques involve Quantitative Surveys, Interviews, Quantitative Observations and Experiments.

The second and most important component is Data Pre-processing. It is very crucial and vital component for Data Science. Typically, the gathered data is disorganized. Errors, missing values, or duplicate entries could be present. Thus, we prepare the data for usage by organizing and cleaning it. This stage contributes to increased precision.

Data Understanding is third component of Data Science. This component try to understand what the data is saying us, we can utilize straightforward visualizations or computations. This facilitates the generation of practical ideas.

Feature Selection is the next component of Data Science which choose the most important data (features) that are useful for solving the problem. Not all data is important, so selecting the right data makes the process better and faster.

After appropriate features selection, model building is to be done. We use methods or algorithms to learn from the data and make predictions or decisions. For example, predicting sales or classifying customers.

Once model is ready, it is evaluated by various evaluation metrics. We check whether it is working correctly or not. If the results are not good, we improve the model.

The next component is Data Visualization where results are shown using charts, graphs, or dashboards so that they are easy to understand. This helps people quickly see the insights.

After everything is ready, the model is used in real life, such as in apps, websites, or business systems. This component is known as Deployment.

The final component is monitoring or improvement. Data Science is a continuous process. We keep checking the system and improve it when needed so that it gives better results over time.

Check your progress-2

a. People and organizations can make better, more informed decisions with the aid of data science (True/False)

b. Through artificial intelligence and machine learning, data science facilitates _____.

c. What are the types of Data Collection?

d. Not all data is important, so selecting the right data makes the process better and faster(True/False)

e. Data Science is a continuous process (True/False).

1.5 Let us sum up

In this unit we have discussed the various practical definitions and their explanation in details. We discussed various evolutions of Data Science from 1960s to current year. We discussed various aspects why Data Science is important. Finally, we discussed the overview of various components of Data Science.

1.6 Check your progress: Possible Answers

1-a

Definition 1: Data Science is the process to better understand problems using data to and make efficient and decisions.

Definition 2: Data Science is the study of gathering, evaluating, and interpreting data in order to derive valuable insights.

Definition 3: Data Science helps organizations use data to make wise and informed decisions.

Definition 4: Data science forecasts future events using data, statistics, and algorithms

Definition 4: Data science forecasts future events using data, statistics, and algorithms.

Definition 5: Data science combines programming, statistics, and subject-matter expertise to extract meaningful data.

Definition 6: Data Science is the study of data using scientific techniques, procedures, and tools to address practical issues.

Definition 7: Data Science is all about learning from data to understand the past, analyse the present, and predict the future.

1-b True

1-c False

1-d Statistics

1-e

- Introduction of Relational Database Management Systems (RDBMS) Structured data storage
- Use of SQL for querying data

1-f Hadoop, NoSQL

2-a True

2-b Automation

2-c Quantitative and Qualitative

2-d True

2-e True

1.7 Further Reading

1. Data Science for Beginners-Compressive Guide to Most Important Basics in Data Science, Alex Campbell, 2021.
2. https://databasetown.com/wp-content/uploads/2019/08/Introduction-to-Data-Science-A-Beginners-guide.pdf

1.8 Assignments

1. Define : Data Science
2. Explain the meaning of definition “Data Science is all about learning from
3. Data to understand the past, analyse the present, and predict the future”
4. Explain any three points why Data Science is important in real lie
5. Explain the modern evolution of Data Science.
6. Explain Feature Engineering, Model Building and Model Deployment components of Data Science.

Unit-2: Categories of Data

Unit Structure

2.0 Learning Objectives

After studying this unit student should be able to:

- Understand various data relevant to data science
- Understand various data types needs
- Aware about Skillsets required for Data Science
- Able to aware about general and well-known issues of various types of Data.

2.1 Introduction

The basis of data science is data. Every model, analysis, and decision-making procedure is dependent upon the kind and caliber of data at hand. Understanding various data types is crucial since the type of data greatly influences the selection of machine learning models, analytical approaches, and visualization strategies.

Data refers to raw facts, figures, or observations collected from various sources. These can be numbers, text, images, audio, or any other form of information that can be processed.

Data generally in raw form and unorganized. It does not provide any meaning without interpretation. Data exist in massive quantity in real life. Data may exist in various forms such as tables means structure form, semi structured like email and unstructured like audio, video, images etc. Basic types of data are describing in following figure

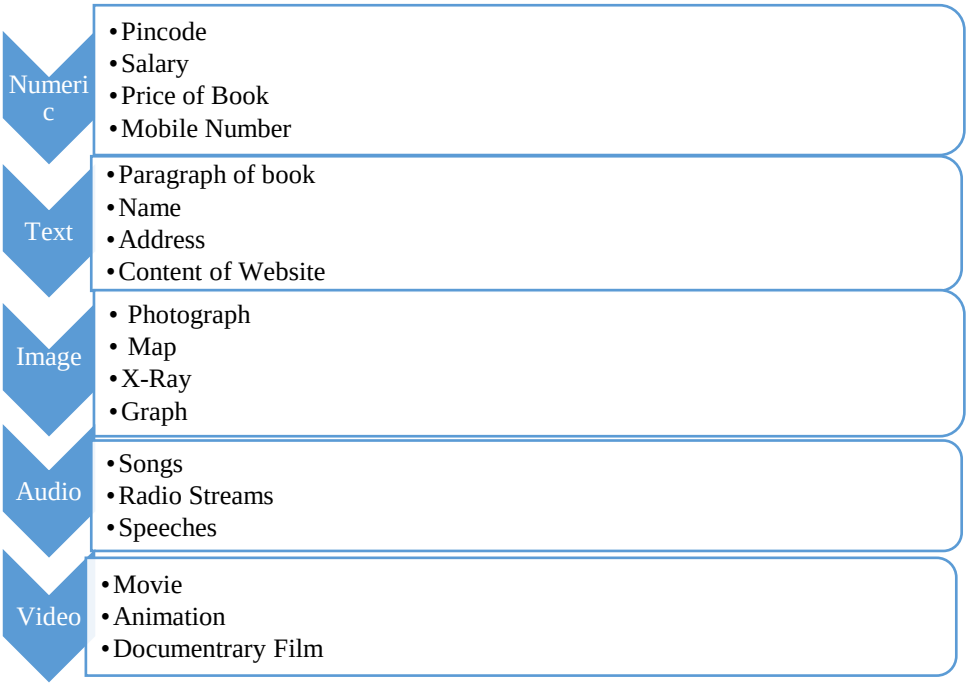


Figure 2.1.1 Types of Data

Data in raw form has no meaning. If it is converted in information form, then it has some logical meaning associated with it. Information is processed, organized, and structured data that provides meaning and context. Data is converted into information through various means such as cleaning, organizing, summarizing and analysing. Information provides context and relevance to the data. It is in organized way so any one can understand it easily. Information is helpful for decision making. Table 2.1.1 describes the difference between Data and Information.

Table 2.1.1 Data Vs Information

Data	Information
• Data is raw material	• Information is product
• Data is unprocessed facts and figures	• Information is processed data
• Data is used as input for computer system	• Information is output of information system
• Data does not depend on Information	• Information depends on data
• Data is not specific	• Information is specific
• Data is a single unit	• Information may be group of data that carries certain meaning
• Data does not carry a meaning	• Information must carry a logical meaning

As far as Data Science is concerned, insights are more useful than information. Insights are deep understanding or conclusions derived from analysing information. They help in making decisions and predictions. Insights are nothing but analysis and interpretation applied on information. Insights are generated through:

- Statistical analysis
- Data visualization
- Machine learning models
- Pattern recognition

2.2 Various Types of Data

In the field of data science, data is most vital entity. There are several ways to classify data according to factors like its measurement scale, composition, and method of collection.

Knowing these kinds is important because the type of data often determines which machine learning models and data analysis methods are used.

Structured Data

Structured data is information that has been arranged in a way that makes it easier to store, search for, and analyse. This arrangement is usually done using rows and columns. It is frequently found in spreadsheets or databases.

- **Main Characteristics**
    - **Tabular Format**: Information is kept in tables that have columns for attributes and rows for records.
    - **Clearly Defined Data Types:** There is a defined data type (such as strings, floats, or integers) for each column.
    - **Easily Searchable:** Structured Query Language (SQL) makes it simple to query structured data.
- **Typical Applications**
        - Hospital Management System
        - Airline Reservation System
        - Institute Management System
        - ERP Systems
        - CRM Systems
        - Telecommunications

Unstructured Data

Compared to structured data, unstructured data is more complex and lacks a predetermined structure, making it more difficult to organize and analyse. Text, pictures, videos, and other types of multimedia content are frequently included.

- **Main Characteristics**
        - **Absence of a Definable Structure**: No set structure or order.
        - **Complexity**: Analysed with sophisticated tools such as computer vision or natural language processing (NLP).
        - **Big in Volume**: Data is frequently kept in big volumes, making indexing and searching more difficult.
- **Typical Applications**
        - Social Media Analysis
        - NLP and Text Mining

- Call Centre Analytics
- Video and Image Recognition
- Email Filtering
- Sentiment Analysis
- IoT and Sensor Data Analysis

Structured Vs Unstructured Data

The table 1.2.1 describes the difference between structure and unstructured data on various parameters.

Table 2.2.1 Difference between Structured Vs Unstructured Data

Parameters	**Structured Data**	**Unstructured Data**
Characteristics	• Predefined Data Models • Usually numbers, text and logical values • Easy to search	• Non Predefined Data Models • Usually text, images, audio, video etc. • Difficult to search
Resides in	• Relational Databases • Data Warehouses	• Applications • NoSQL databases • Data Warehouses • Data Lakes
Generated By	Human or machine	Human or machine
Typical Applications	• Hospital Management System • Airline Reservation System • Institute Management System • ERP Systems • CRM Systems	• Social Media Analysis • NLP and Text Mining • Call Centre Analytics • Video and Image Recognition • Email Filtering • Sentiment Analysis • IoT and Sensor Data Analysis
Examples	• Date • Mobile Number • Adhar Number	• Text files • Reports • Email messages

	• Name • City • Product id	• Audio files • Video files • Images

Semi-structured Data

Between structured and unstructured data is semi-structured data. Even though it doesn't follow a strict schema, it still has tags or markers to distinguish between various elements. Although semi-structured data still lacks the formal structure of relational databases, it is more organized than unstructured data.

- **Main Characteristics**
  - **Flexibility**: Easier to interpret and evaluate than unstructured data, but more adaptive than structured data.
  - **Storage**: Frequently kept in specialized file formats like JSON or NoSQL databases.
- **Applications**
  - Web Scraping
  - Sentiment analysis
  - Spam Detection
  - Parsing of E-mail
  - No SQL Database based Applications
  - Smart Homes
  - Social Network Analysis

Quantitative Data

Data that is measurable or countable is referred to as quantitative data. It is primarily utilized in statistical modelling, hypothesis testing, and different machine-learning algorithms due to its numerical nature.

One can further categorize quantitative data into:

- **Discrete Data**: Discrete data are those that have a limited range of possible values. These values are distinct, countable, and frequently limited. Put differently, discrete data can only contain whole numbers; values in between the numbers are not possible. It is usually applied to variables for which there are only distinct values and no possibility of intermediate values. For example, *a class's number of students is countable and cannot be expressed in fractions.*

- **Continuous Data:** Any value within a given range can be found in continuous data. These values are measurable, and there is always an infinite range of possible values between any two values. Measurements as opposed to counts are frequently the source of continuous data. For example, a person's height can range from 4.8 feet to 6.3 feet, or any combination of the two.

The table 2.2.2 describes the difference between Discrete and Continuous data in terms of number of parameters.

Table 2.2.2 Difference between Discrete and Continuous Data

Aspects	**Discrete**	**Continuous**
Characteristics	Countable, distinct values	Measurable, infinite possible values
Values	Integers	Any values. Including fractions and decimals values
Intervals	Fixed	Infinite
Statistical Methods	Binomial, Poisson	Normal Distribution
Measurement Methods	Counting	Measurement
Visualization Methods	Bar Chart, Pie Chart	Line Graphs, Histograms
Examples	Number of Staff members, Number of Students	Temperature, Time, Height, Weight

Qualitative Data

Data that is not numerical and describes attributes or features is referred to as qualitative data. It is used to classify objects and is especially helpful in disciplines like the social sciences where it is crucial to comprehend human behaviour or opinions.

One kind of qualitative data that depicts groups or categories is called categorical data. It may be ordinal or nominal:

- **Nominal Data**: Discrete categories without a natural order are represented by nominal data. For instance, different kinds of cars (SUV, sedan, hatchback) or fruit (banana, orange, and apple).
- **Ordinal Data:** Ordinal data is a set of categories with a clear hierarchy but no discernible separation between them. For instance, the star ratings for movies

(1, 2, or 3 stars) or the degrees of education (high school, undergraduate, graduate).

Important distinctions between qualitative and quantitative data

While qualitative data is descriptive and used for classification, quantitative data is numerical and used for exact computations. While statistical models and numerical techniques are frequently used to analyse quantitative data, text analysis and sentiment analysis are two possible methods for analysing qualitative data.

Big Data

Massive data sets that are too big to handle with conventional data processing methods are referred to as "big data." Big data is frequently defined by the following five V's:

- **Volume:** The sheer amount of data being produced and kept. The size of data is not fixed. However, data with large volume
- **Variety:** The range of data kinds, encompassing semi-structured, unstructured, and structured data.
- **Velocity:** The rate at which information is produced and analysed. For instance, information from social media sites or Internet of Things sensors.
- **Veracity:** The data's quality or dependability, which can be difficult to guarantee with big and varied data sets.
- **Value:** The information that can be gleaned from the data that is practical or insightful.

Check Your Progress-1

a. Differentiate Structured and Unstructured data.

b. Semi-structured data is easier to interpret and evaluate than unstructured data, but more adaptive than structured data (True/False).

c. Data that is measurable or countable is referred to as _____ (Quantitative/ Qualitative).

d. Number of staff in the institute is an example of _____ (Discrete/Continuous)

e. The star ratings for movies is an example of _____(Nominal/ Ordinal)

F._____ characteristic of Big Data determines the rate at which Information is produced and analysed.

(i)Volume (ii) Variety (iii) Velocity (iv) Veracity

2.3 Different Datatypes for Data Science

Understanding data types is crucial to data science since various data kinds call for various approaches to processing, storing, visualizing, and analysing data. The type of data greatly influences the selection of appropriate methods, algorithms, and instruments.

Categorical data represents characteristics or labels rather than numerical values. It is used to group data into categories. It is divided into two main categories: Nominal and Ordinal. Nominal data consists of categories without any order or ranking. One of the most basic categories of data in data science is nominal data, which falls under the more general heading of qualitative or categorical data. It describes data that is made up of names or labels used to distinguish several groups without any sort of natural ranking or order. There is no notion of size, sequence, or hierarchy with nominal data; each value only denotes a unique class. Categories like blood group (A, B, AB, O), gender (male, female), and city names, for instance, are all notional. In these situations, it is illogical to arrange the values in ascending or descending order because they are only descriptive. Ordinal data is a kind of category (qualitative) data where the distinctions between the categories are not quantifiable or consistent, but the values indicate categories with a clear and meaningful order or ranking. To put it another way, ordinal data does not quantify how much one category is higher or lower than another; rather, it only indicates the relative location of values, which is either higher or lower. Because of this, ordinal data differs conceptually from numerical data, which contains quantifiable differences, and nominal data, which has no order.

Numerical or Qualitative is another important data type used in Data Science. Numerical data represents measurable quantities and allows mathematical operations. It is divided into two main parts: Discrete and Continuous. Numerical data made up of countable, separate values is referred to as discrete data. These values are typically whole numbers (integers) and arise from counting rather than measuring. The absence of intermediate values between two successive observations is a crucial characteristic of discrete data. For example, the number of students in a classroom can be 40 or 41, but not 40.5. Similarly, the number of cars in a parking lot, number of defects in a product, or number of customers visiting a store are all discrete in nature.

Continuous data, on the other hand, describes numerical data that can have any value within a specified range, including decimal and fractional values. Continuous data provides for unlimited precision and is produced by measuring as opposed to counting. For instance, because they might have values like 170.2 cm, 65.75 kg, or 36.6°C, height, weight,

temperature, time, and distance are all continuous variables. There are an endless number of possible intermediate values between any two numbers.

Binary datatype is also another important data type used in Data Science. Binary data is a special type of categorical data with only two possible values. It can be narrated as yes/no, true/false, pass/fail. It is used in classification problems of Data Science.

Time based datatype is also widely used in Data Science. It is collected at regular intervals over time.

Text datatype is also vital in Data Science which is used various applications of Data Science like Email, Social Media posts, reviews etc.

Other important data types are audio, video, images, geospatial, graph etc.

2.4 Importance of understanding Data Types

The success of data analysis, modelling, and decision-making is strongly impacted by an understanding of data types, which is more than just a fundamental idea in data science. Every dataset is made up of various data types, each of which has a distinct behaviour. Inaccurate analysis, deceptive findings, and subpar model performance may emerge from a data scientist's failure to recognize and manage these kinds. Therefore, at every level of the Data Science lifecycle, a thorough grasp of data types is crucial.

Understanding data types is crucial for choosing the right analytical and machine learning methods. Various algorithms are made to handle different kinds of data. For instance, classification algorithms are applied to categorical data, but regression methods are appropriate for continuous numerical data. The model may perceive relationships improperly and produce biased or inaccurate predictions if categorical input is mistakenly handled as numerical without appropriate encoding. Therefore, applying the appropriate algorithm to the appropriate problem is ensured by knowing the data type.

Data Pre-processing and transformation, which is an essential step prior to any analysis or modelling, is another important component. Real-world data frequently has to be cleaned, formatted, and transformed because it is untidy. While categorical data must be translated into numerical form using methods like label encoding or one-hot encoding, numerical data may require scaling or normalization. Tokenization and vectorization are also necessary for text data. Applying the proper pre-processing techniques is impossible without knowing the type of data, which can have a major impact on the quality of the findings.

Statistical analysis and interpretation is another crucial component. The type of data determines which statistical tests and measures are used. The mean and standard deviation, for instance,

have significance for continuous data but not for nominal data. It is preferable to use rank-based or median analysis for ordinal data. Similarly, non-parametric tests are appropriate for ordinal or non-normal data, whereas parametric tests are employed for numerical data under specific assumptions. Decision-making may be impacted by inaccurate statistical findings resulting from a misinterpretation of data kinds.

Understanding data types is essential for feature engineering and model performance in machine learning. To increase model accuracy, features derived from various data formats must be handled differently. For instance, it could be necessary to encode categorical variables, scale continuous variables, and extract features like trends or seasonality from time-series data. When data types are handled correctly, prediction performance is improved, mistakes are decreased, and model efficiency is increased.

Check your progress-2

a. _____ data represents characteristics or labels.

b. Nominal data consists of categories without any order or ranking(True/False)

c. _____data is a special type of categorical data with only two possible values.

d. Why understanding of data is important?

2.6 Let us sum up

In this unit we have discussed various types of data in details. The data types used for Data Science is discussed in the details. The unit also discussed the importance of data types in Data Science stages.

2.7 Check your progress: Possible Answers

1-a

Parameters	**Structured Data**	**Unstructured Data**
Characteristics	• Predefined Data Models • Usually numbers, text and logical values • Easy to search	• Non Predefined Data Models • Usually text, images, audio, video etc. • Difficult to search

Resides in	• Relational Databases • Data Warehouses	• Applications • NoSQL databases • Data Warehouses • Data Lakes
Generated By	Human or machine	Human or machine
Typical Applications	• Hospital Management System • Airline Reservation System • Institute Management System • ERP Systems • CRM Systems	• Social Media Analysis • NLP and Text Mining • Call Centre Analytics • Video and Image Recognition • Email Filtering • Sentiment Analysis • IoT and Sensor Data Analysis
Examples	• Date • Mobile Number • Adhar Number • Name • City • Product id	• Text files • Reports • Email messages • Audio files • Video files • Images

1-b Quantitative

1-c Discrete

1-d Ordinal

1-e velocity

2-a Categorical

2-b True

2-c Binary

2-d The success of data analysis, modelling, and decision-making is strongly impacted by an understanding of data types, which is more than just a fundamental idea in data science. Every dataset is made up of various data types, each of which has a distinct behaviour. Inaccurate analysis, deceptive findings, and subpar model performance may emerge from a

data scientist's failure to recognize and manage these kinds. Therefore, at every level of the Data Science lifecycle, a thorough grasp of data types is crucial.

Understanding data types is crucial for choosing the right analytical and machine learning methods. Various algorithms are made to handle different kinds of data. For instance, classification algorithms are applied to categorical data, but regression methods are appropriate for continuous numerical data. The model may perceive relationships improperly and produce biased or inaccurate predictions if categorical input is mistakenly handled as numerical without appropriate encoding. Therefore, applying the appropriate algorithm to the appropriate problem is ensured by knowing the data type.

Data Pre-processing and transformation, which is an essential step prior to any analysis or modelling, is another important component. Real-world data frequently has to be cleaned, formatted, and transformed because it is untidy. While categorical data must be translated into numerical form using methods like label encoding or one-hot encoding, numerical data may require scaling or normalization. Tokenization and vectorization are also necessary for text data. Applying the proper pre-processing techniques is impossible without knowing the type of data, which can have a major impact on the quality of the findings.

Statistical analysis and interpretation is another crucial component. The type of data determines which statistical tests and measures are used. The mean and standard deviation, for instance, have significance for continuous data but not for nominal data. It is preferable to use rank-based or median analysis for ordinal data. Similarly, non-parametric tests are appropriate for ordinal or non-normal data, whereas parametric tests are employed for numerical data under specific assumptions. Decision-making may be impacted by inaccurate statistical findings resulting from a misinterpretation of data kinds.

Understanding data types is essential for feature engineering and model performance in machine learning. To increase model accuracy, features derived from various data formats must be handled differently. For instance, it could be necessary to encode categorical variables, scale continuous variables, and extract features like trends or seasonality from time-series data. When data types are handled correctly, prediction performance is improved, mistakes are decreased, and model efficiency is increased.

2.8 Further Reading

1. https://www.geeksforgeeks.org/data-analysis/what-is-data/
2. https://www.geeksforgeeks.org/maths/data-types-in-statistics/
3. https://www.blog.trainindata.com/data-science-fundamentals/

2.9 Assignments

1. Give any two examples of image, audio and video data.
2. Differentiate Data and Information.
3. What do you mean by insight?
4. Differentiate structure and unstructured data.
5. What are the categories of qualitative data? Explain.
6. What data types is important for data pre-processing.

Unit-3: The Data Science Process

Unit Structure

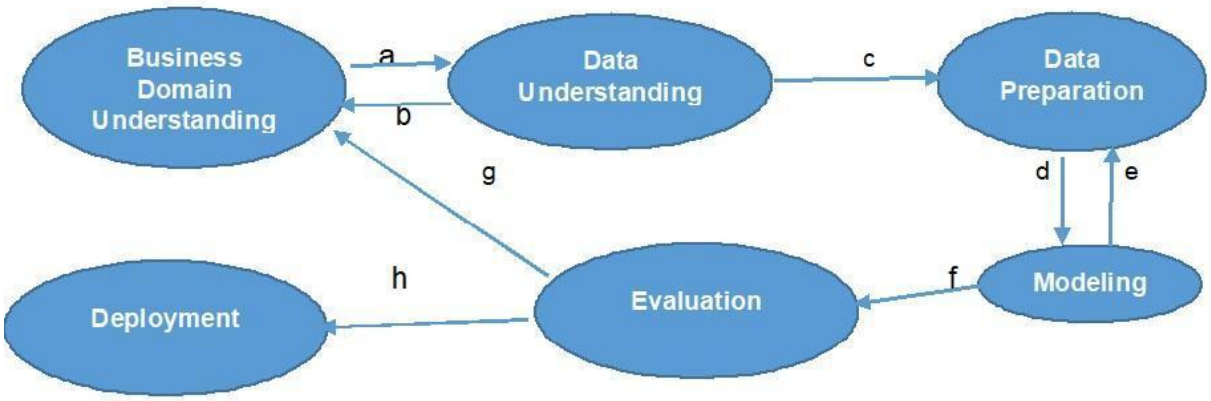
3.0 Learning Objectives

After studying this unit student should be able to:

- Know stages and objective of each stage
- Able to understand various aspects to understand data and business
- Able to understand objectives, advantages and steps of Data Preparation
- Able to understand various kind of problems and associated algorithms.
- Able to understand objective of Evaluation and Deployment phases of Data Science

3.1 Introduction

Implementation of effective Data Science process is decisive for the success of any Data Science project. The Data Science process comprises of various steps depicted as Figure 3.1.1



A Needs of Business **b** Context to **c** Appropriate and Purity of Data

D Techniques **e** Improvisation of Data **f** Business objectives

G Results

(Figure 3.1.1 Stages of Data Science Process)

Data Science Process starts with understanding of business domain. It generally tries to identify what business need. Data Scientist at this stage strive to understand business problem. A thorough grasp of the business domain guarantees that the issue is pertinent to the objectives of the organization, whether it is to reduce costs, revenue growth, customer satisfaction enhancement, etc. Data Scientist understands business's key performance indicators (KPIs) at this stage. It helps Data Scientist to choose which data features to incorporate into proposed

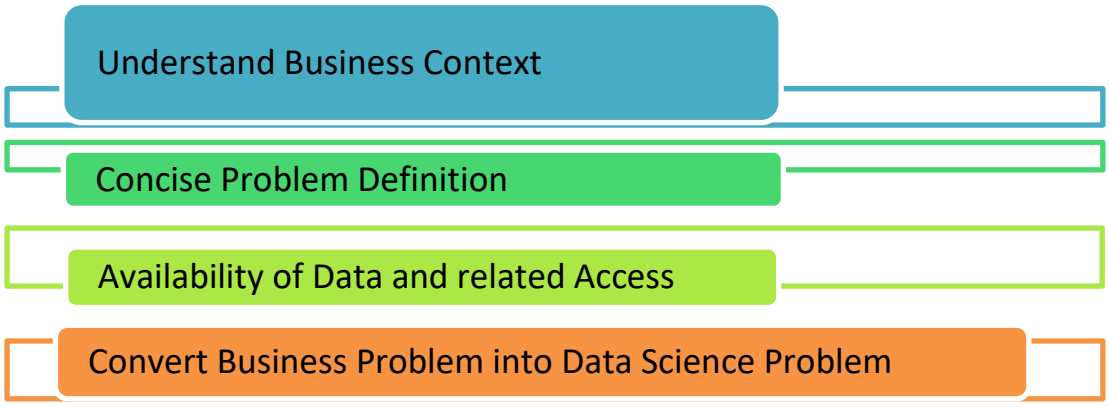
models. After business understanding, Data Scientist strives to understand data. It gathers information about availability of data and what data is needed. At this stage, Data Scientist also check the quality of data. Data Scientist decides what types of pre-processing is required and what data is to be pre-processed. The next stage is Data Pre-processing. In data science, data pre-processing is an essential step that guarantees raw data is converted into a clear and comprehensible format for analysis and modelling. The process of using appropriate data to create a mathematical representation of real-world phenomena or processes is known as data modelling. Building models that can recognize patterns, anticipate outcomes, and deduce relationships from data is the practice of data science. This stage, which comes after data pre-processing, is crucial for deriving insightful conclusions and creating statistical or machine learning models. After modelling, the next stage is evaluation. It is one of the crucial stage in the data science lifecycle. Models are evaluated at this point to see how well they work, how accurate they are, and how well they can generalize to new data. Making sure the model performs well on both training and unseen data is the main objective. Understanding whether a model is appropriate for deployment in a real-world scenario requires a thorough evaluation. The last step in the data science process is called "model deployment," when a statistical or machine learning model that has been trained is integrated into a real-time production environment to make predictions or offer insights on fresh data. The deployment phase encompasses more than just the technical implementation of a model; it also entails maintaining the model's durability, dependability, scalability, and efficiency over time.

3.2 Business and Data Understanding

The figure 3.2.1 depicts main steps for Business understanding in Data Science process. The first step of Business Understanding is to formulate business problem efficiently. In data science, a business problem usually involves using data to solve practical business problems or streamline operational procedures. It's crucial to adhere to structured procedures and approaches that convert the business challenge into a data science problem in order to approach this methodically.

(Figure 3.2.1 Steps of Business Understanding)

The figure 3.2.2 describes vital steps to formulate business problem in context to Data Science.



(Figure 3.2.2 Vital Steps to formulate Business Problem)

The central objective to understand business context is exhaustive considerate of business domains, objectives and key contests. Some questions are very important for Data Scientist in

order to understand business domains, goals and key challenges. Data Scientist should ask several questions such as:

- What does the company hope to accomplish?
- Which inefficiencies or problems exist in the current system?
- What value can data science provide?

The second step is to clearly and concisely define the problem on the basis of several questions.

- What particular issue are we resolving?
- What quantifiable results, or key performance indicators (KPIs), are there?

The third step is very crucial for Data Scientist as data plays a very important role in any Data Science project. Determine the sources of the data and evaluate the data's accessibility, accuracy, and applicability is the main role of Data Scientist.

The fourth and final vital step is to convert business problem into Data Science problem. Understanding the underlying goals of the business and translating them into a format that data science can handle are necessary steps in converting business problems into data science problems. For example:

Business Problem: A retail organization wants to lower customer attrition in order to boost revenue.

Data Science Problem: Identify the customers who are most likely to leave in the next six months.

The business objectives that are established during the business understanding phase are very important in data science because they serve as the project's overall direction. Understanding the objectives of the company and how data can support them is the aim. Data scientists can maximize the value of data-driven decision-making by customizing their analysis and model-building processes to deliver insights that directly address the needs of the business by having a thorough understanding of the business objectives.

The third phase of business understanding is to frame right questions. Asking the right questions during the business understanding phase of a data science project is essential to elucidating the goals, limitations, and scope. These inquiries aid in ensuring that data science initiatives complement business requirements. The important questions are as follows:

- Which is the main opportunity or problem for business?
- What particular business objectives are there?
- What are the success-related Key Performance Indicators (KPIs)?
- Which underlying business processes are at play here?

- What are the roles of the stakeholders, and who are they?
- What limits or restrictions exist?
- What information is out there, and what format is it in?
- What is the anticipated time frame for results delivery?
- What kind of integration will the data science solution make with the existing systems?
- What hazards exist and how will they be addressed?
- What kind of resources or assistance are offered?

A data science project's business understanding phase's success metrics centre on how well the project fits with organizational objectives and lays the groundwork for insightful analysis. These metrics aid in determining whether the project is well-positioned for success and whether the problem is well-understood.

To understand business thoroughly, feasibility and constraints play an important role. Feasibility of Data Availability, Technological infrastructure, Skills of Team, Domain expertise needs to be determined effectively. The constraints should also need to be identified. Time, Cost, Resources, Communications constraints are vital for the business understanding.

Data Scientist should consider business priorities while understanding business and data. Data Scientist gives priority to different tasks while understanding business.

3.3 Data Preparation

One of the most important steps in the data science pipeline is data preparation. It entails gathering, sanitizing, converting, and arranging unprocessed data into a structure appropriate for modelling or analysis. This stage makes sure the data is correct, comprehensive, and prepared for efficient use in the data science procedure. Data preparation is vital and it has numerous advantages such as:

Enhances Data Quality: Inconsistencies, errors, missing values, and irrelevant information are frequently found in raw data. These problems are resolved by data preparation methods like cleaning, imputation, and normalization, which produce a dataset that is more streamlined and consistent. This therefore stops these problems from biasing or impeding your models' learning process.

Improves Model Performance: The quality of the data used to train machine learning algorithms is a major factor in this process. Effective data preparation gives the algorithms a clean, organized base on which to discover patterns and connections. As a result, models become more generalizable and predictive of unseen data.

Save Time and Resources: Devoting time up front to data preparation can result in substantial time and resource savings later on. Early resolution of data quality concerns helps you avoid problems later in the modelling process that may call for troubleshooting or rework. This results in a machine learning workflow that is more streamlined and effective.

Data Preparation generally contains following steps:

1. Dataset import and export
2. Changing the name of a variable
3. Modifying the kind of variable
4. Sorting using duplicate keys or whole duplicate records for one or more variables
5. Picking columns to add to the output dataset from the input dataset
6. Rows are filtered according to one or more criteria
7. Developing new variables using functions for already-existing variables
8. Conditional variable processing
9. Tables Appendices
10. Creating join tables (complete outer join, left and right join, inner join)
11. Switch around the tables
12. A summarize column that is broken down by groups
13. Standardizing and normalizing columns (with respect to continuous variables)
14. Continuous variable betting
15. Entering values that are missing into a variable

Check your progress-1

a) List out steps of Data Science Process.
b) Which questions are very important for Data Scientist in order to understand business domains, goals and key challenges?
c) Give advantages of Data Preparation.

3.4 Data Modelling

Data Modelling is crucial step in Data Science. It is to be done with the help of algorithms. The main challenge for Data Scientist is to decide when to use which algorithm in real life. Generally, there are five kinds of problems exist in real life. The decision of algorithm is to be done on the basis of the nature of the problem.

[1] Is this X or Y kind of Problem: These kind of problems have categorical answers? The answers are fixed as per predefined classes. For example: Arrange the students' marks data in Distinction, First Class, Second Class, Pass Class and Fail.

The predefined classes are:

Distinction: Marks >=80 and <=100

First Class: Marks >=75 and <=79

Second Class: Marks >=70 and <=74

Pass Class: Marks >=50 and <=69

Fail: Marks <50

In such kind of problems, classification algorithms are used.

[2] How many and how much kind of problem: When output of the problem contains figures or numerical values then this kind of problems are known as how many or much kind of problems.

For Example: What will be the temperature for tomorrow? Based on available existing data

Temp = [8, 7, 8, 6, 7, 8]

In such kind of problems, regression algorithms are used.

[3] How is this organized kind of problems?

Let's say you have some data, but you have no idea how to interpret it. Well, clustering algorithms can help you solve it. Algorithms for clustering group data according to shared characteristics.

For example: We possess the percentage attendance data for M.Sc. IT students in semester I. Arrange this data into groups with similar characteristics?

Attendance = [85,60,82,75,45,85,90,82,71,56,47,58,75,95]

In such kind of problems, clustering algorithms are used.

[4] Is this weird kind of problem?

When in the datasets, unusual pattern identifies then that kind of problem is of this category. For example, assume that the vector contains the marks of particular students in all courses of 1st Semester M.Sc.IT out of 30, try to determine weird pattern or anomaly in the same.

Marks = [26, 27, 28, 8, 29]

In such kind of problems, anomaly detection kind of techniques are used.

[5] What Should I do next kind of Problem?

Based on the training, when system or computer takes decision automatically then those problem considered as of this category. Basically reinforcement learning kind of algorithms used in this category.

As an illustration, consider your temperature control system's decision to either raise or lower the room's temperature.

3.5 Evaluation and Deployment

The Evaluation Phase makes sure the model satisfies both statistical performance goals and business objectives. The Deployment Phase guarantees the model's successful integration into production systems. During the evaluation phase, the machine learning models' dependability and performance are evaluated to make sure they fulfil the project's goals.

Different performance metrics are needed for different kinds of models in evaluation phase. Typical measurements consist of: Classification criteria include ROC-AUC (Area under the Receiver Operating Characteristic Curve), F1 Score, Accuracy, Precision, and Recall. Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R2 Score are the metrics used in regression analysis. Silhouette Score, Davies-Bolden Index, and Adjusted Rand Index are used for clustering. Precision, Recall, F1 Score, and Area under Precision-Recall Curve (AUPRC) are used in anomaly detection. In evaluation phase, It is also to be checked that model is overfitting to the training data (performing well on training but poorly on test data) or under fitting (performing poorly on both training and test data). In this phase, it is also evaluate whether the model’s results align with business goals or not.

In Deployment phase, the machine learning model is integrated into the production environment so that it can communicate with actual users and data. Through the automation of choices or insights, this phase guarantees that the model continuously delivers value. Various steps are needed in this phase such as Model Integration, Containerization, Automation and Scheduling, Monitoring and Maintenance, Security consideration.

Check your progress-2

(a)Which kind of algorithms are used to predict whether student will pass or fail?

(b) Which kind of algorithms are used to predict next semester CGPA of students based on past data?

(c) What is the objective of evaluation and deployment stage of Data Science?

3.6 Let us sum up

In this unit we have discussed stages of Data Science in details. Unit started with understanding of data and business. The unit discussed the steps to understand data and business effectively. The unit discussed the advantages and various steps of data preparation. Data modelling is the vital step of Data Science. The unit discussed various categories of problems and associated algorithms. Finally, the unit discussed the objective and actions of evaluation and deployment phases of Data Science.

3.7 Check your progress: Possible Answers

1-a Steps are (1) Business Domain Understanding (2) Data Understanding (3) Data Preparation (4) Data Modelling (5) Evaluation and (6) Deployment

1-b (1) What does the company hope to accomplish? (2)Which inefficiencies or Problems exist in the current system? (3) What value can data science Provide?

1-c Advantages are :

Enhances Data Quality: Inconsistencies, errors, missing values, and irrelevant information are frequently found in raw data. These problems are resolved by data preparation methods like cleaning, imputation, and normalization, which produce a dataset that is more streamlined and consistent. This therefore stops these problems from biasing or impeding your models' learning process.

Improves Model Performance: The quality of the data used to train machine learning algorithms is a major factor in this process. Effective data preparation gives the algorithms a clean, organized base on which to discover patterns and connections. As a result, models become more generalizable and predictive of unseen data.

Save Time and Resources: Devoting time up front to data preparation can result in substantial time and resource savings later on. Early resolution of data quality concerns helps you avoid problems later in the modelling process that may call for troubleshooting or rework. This results in a machine learning workflow that is more streamlined and effective.

2-a Classifications

2-b Regressions

2-c The Evaluation Phase makes sure the model satisfies both statistical performance goals and business objectives. The Deployment Phase guarantees the model's successful integration into production systems

3.8 Further Reading

1. Bad Data Handbook: Cleaning Up The Data So You Can Get Back To Work, Q. Ethan McCullum , 2012
2. https://www.geeksforgeeks.org/what-is-data-preparation/

3.9 Assignments

1. List out stages of Data Science Process and explain Business and Data understanding stage.
2. What is the objective of Data Preparation? Give advantages and steps of it.
3. Explain is this x or y problem with example.
4. Explain What Should I do next kind of Problem.
5. Explain objectives of evaluation and deployment phases of Data Science.

Unit-4: Applications and Tools of Data Science

Unit Structure

4.0 Learning Objectives

After studying this unit student should be able to:

- Understand various applications where Data Science concepts are used extensively
- Able to aware about popular Data Science Programming languages and their features
- Able to aware about popular Data Science Libraries and Framework.

4.1 Introduction

Data is created at a very rapid pace in today's digital world from a variety of sources, including websites, social media, mobile applications, sensors, and business transactions. Data is generated and kept each time we use the internet, make an online purchase, watch a movie, or even use navigation systems to travel. When properly examined, this vast amount of data can yield insightful information. Data science is crucial in this situation. In order to analyse data and derive valuable insights, the multidisciplinary area of data science integrates ideas from computer science, mathematics, and statistics. By revealing patterns, trends, and linkages concealed inside data, it aids both individuals and businesses in making better decisions. But data science is more than just data analysis; it also entails implementing strategies and utilizing appropriate tools to tackle real-world issues.

The numerous applications of data science are among its most significant features. Nearly every industry, including healthcare, banking, education, agriculture, transportation, and entertainment, uses data science. For instance, banks use it to identify fraud, hospitals use it to forecast illnesses and enhance patient care, and e-commerce businesses use it to make product recommendations. These uses demonstrate how data science is enhancing daily life and changing industry.

In data science, using the right tools and technology is just as crucial as applications. Data collection, storage, processing, analysis, and visualization are all aided by these tools. Data science frequently makes use of programming languages like Python and R, data analysis tools like spreadsheets, visualization tools like dashboards, and machine learning frameworks. Every tool has a distinct function, and the type of data and the issue being resolved determine which tool is best. Understanding the applications and tools of Data Science is essential for all. It helps people to connect theoretical concepts with real-world usage and prepares them for careers in data-related fields. Learning about tools also provides practical skills that are highly demanded in the industry.

4.2 Applications of Data Science

Data Science plays a vital role in every business now a day. It assists businesses in identifying trends, analysing vast volumes of data, and coming to wise judgments. Data science has a wide range of applications that are always growing as technology advances.

- **Healthcare**: One of the sectors where data science is having the biggest influence is healthcare, where it is changing how illnesses are identified, treated, and avoided. Data science assists medical professionals in making quicker, more precise, and customized decisions due to the growing availability of medical data, including genetic data, wearable device data, electronic health records, and medical imaging. Predicting illnesses before they worsen is one of the most significant uses of data science in healthcare. To find risk variables, machine learning algorithms examine patient data like age, medical history, lifestyle, and lab results. Medical Image Analysis is another important application of healthcare of Data Science. Large volumes of data are produced by medical imaging, including MRIs, CT scans, ultrasounds, and X-rays. These photos can be analysed with the aid of data science tools, particularly deep learning. Personalized Medicine is vital application of health care of Data Science in health care field. Personalized treatment according to a patient's unique traits, such as genetics, lifestyle, and medical history, is known as personalized medicine. Drug Discovery and Development is another important and vital application of health care of Data Science. The process of creating new medications is costly and time-consuming. Through the analysis of biological data and the identification of possible medication candidates, data science expedites this process. Patient Monitoring and Wearable Devices application is emerging application of Data Science in health care to monitor patient health in real time. Telemedicine and remote health care is also emerging application of health care of Data Science. It allows patients to consult doctors remotely using digital platforms.
- **Business and E-Commerce**: Modern company and e-commerce operations now heavily rely on data science. Businesses produce enormous volumes of data via supply chains, mobile apps, websites, transactions, and customer interactions. Businesses may make smarter decisions, boost revenue, and provide better customer service by studying this data. Businesses that use customer data wisely, such as Amazon and Flipkart, are excellent examples of how data science promotes commercial success. Customer Segmentation, Recommendation Systems, Sales Forecasting, Inventory Management,

Sentiment Analysis and Customer Feedback, Churn Prediction are major applications of Data Science in this field. Customer segmentation is the process of grouping consumers according to shared attributes like age, region, behaviour, or buying habits. Main applications of this category are: Personalized marketing campaigns, Targeted advertisements, Product recommendations etc. Sales forecasting application predicts future sales based on historical data and market trends. Inventory management is another important category of this field. Effective inventory management is essential for e-commerce companies. The proper balance between supply and demand is maintained with the use of data science. Sentiment Analysis and Customer Feedback is yet another important application of this field. As per this application, businesses examine social media comments, reviews, and ratings. Churn prediction identifies customers who are likely to stop using a service.

- **Finance and Banking:** Data science is used by financial institutions to identify fraudulent transactions and control risks. Banks analyse transaction data to identify suspicious activities and prevent financial crimes. Credit Scoring, Risk Analysis, Algorithmic trading, Fraud Detection Systems are examples of Data Science applications in this field.
- **Education:** Data Science plays a vital role in education sector. Data science is used by educational institutions to enhance learning outcomes and customize instruction. Learning platforms make personalized learning path recommendations by analysing student performance data. Best applications of Data Science in education are: Student Performance Analysis, Adaptive Learning Systems, and Course Recommendation etc.
- **Social Media and Entertainment:** Today's era of social media and Data Science play very important role in this field as well. Data science is used by social media companies like Facebook and Netflix to assess user activity and offer tailored content recommendations. Content Recommendation, Sentiment Analysis, Trend Analysis are best examples of Data Science in social media and entertainment.
- **Transportation and Logistics:** Data science enhances delivery efficiency, lowers fuel use, and optimizes routes in transportation and logistics industry. Uber and other companies utilize data analytics to effectively match drivers with passengers. Best examples of Data Science in this area are: Route Optimization, Traffic Prediction and Fleet Management.

Check your progress-1

(a) Nearly every industry, including healthcare, banking, education, agriculture, transportation, and entertainment, uses data science (True/False)

(b) Give attest two examples of Data Science applications in Finance and Banking sector.

(c) How Data Science is used in Social Media and Entertainment?

(d) What is the role of Data Science in Education?

4.3 Programming Languages

There are various tools and technologies used in Data Science for variety of tasks. Two main popular programming languages used in Data Science are Python and R.

Python: Due to its readability, ease of use, and a wealth of libraries such as Pandas for data manipulation, NumPy for numerical calculations, Matplotlib and Seaborn for visualization, and Scikit-learn for machine learning, Python is the most widely used language in data science. Its adaptability enables scalability to intricate projects and rapid prototyping.

R: Originally created for statistical computing, R is still crucial to data science, particularly in research and academia. It has strong statistical modelling tools and excels at data visualization with packages like ggplot2 and plotly. R is perfect for data visualization and in-depth statistical analysis.

In addition to this, SQL, Julia, SAS, Java and Scala languages are also used widely in Data Science.

Together, these languages provide the foundation of data science, with each language offering special advantages. While SQL and Java-based languages are essential for processing and querying big data, Python and R continue to be the most popular due to their ease of use and vast ecosystems. As the field develops, data scientists frequently select languages according to project requirements, combining several languages to capitalize on each one's advantages.

Python's ease of use, readability, and large library support make it the most popular programming language in data science. The field of data science includes gathering, cleaning, analyzing, visualizing, and creating predictive models from data. Python is the ideal tool for every step of this process because of its adaptability, flexibility, and dynamic nature. Because of the language's simple syntax, data scientists of all skill levels can write, comprehend, and

maintain code with ease. Furthermore, Python's popularity in data science is a result of both its user-friendliness and its active community, which has produced and maintained robust libraries designed especially for machine learning and data science.

Advantages of Python

Readability and Simplicity: Python's syntax is similar to that of natural language, which makes it easier for novice data scientists to learn. This simplicity facilitates quicker development and lowers the likelihood of code errors.

Entire Ecosystem: Python offers complete solutions for data science workflows with libraries like Scikit-Learn, Pandas, and TensorFlow. Data scientists can handle different aspects of the data science process without switching to a different language.

Strong Community Support: Python boasts a sizable developer and data scientist community that actively participates in open-source projects, which facilitates the discovery of resources, guides, and answers to possible issues.

Platform Independence: Python can be used in any environment because it is compatible with Windows, macOS, and Linux.

Because of its origins in statistical computing, the **R programming** language is essential to data science, especially statistical analysis and data visualization. R, which was initially created for statisticians, is highly valued by data scientists for its potent modeling, data manipulation, and visualization features. In fields like academia, research, and the social sciences, healthcare, and finance that demand in-depth statistical analysis, it is frequently the preferred language.

R's specialized packages and functions, which facilitate smooth data exploration, visualization, and statistical modeling, are the foundation of its popularity in data science. For data professionals with a background in statistics, its syntax is very accessible because it is made to be intuitive for statisticians. Thousands of packages covering everything from fundamental data manipulation to sophisticated machine learning are available in R's extensive library ecosystem on CRAN (Comprehensive R Archive Network).

Advantages of R in Data Science

Specialized for Statistics and Data Analysis: R has a clear advantage in fields requiring intricate statistical calculations because it was created especially for statistical analysis. Because its functions and syntax are designed with statisticians' needs in mind, data analysts in these domains will find it easier to use.

Vast Library Ecosystem: R's libraries encompass almost all facets of statistical modeling and data science. R is a versatile tool for many fields because it comes with packages for time series, econometrics, biostatistics, genetics, and more.

Reporting and Visualization: R has some of the best visualization features available, and ggplot2 is a benchmark for data visualization. Additionally, R Markdown and Shiny facilitate the creation of interactive applications and comprehensive reports, which facilitate the dissemination of insights to stakeholders.

Community and Support: Researchers, statisticians, and data scientists all contribute to R's lively and encouraging community. Its community upholds strict guidelines for packages, making sure that updates and documentation maintain the ecosystem's stability and usability.

Academic and Research Preference: R is perfect for cutting-edge statistical research and specialized analyses because of its widespread use in academic and research settings, which guarantees that a variety of complex methodologies and algorithms are implemented first in R.

4.4 Data Science Libraries and Frameworks

Data science libraries offer crucial tools that streamline and improve data scientists' workflows, making it simpler to process, analyze, and visualize data as well as create machine learning models. NumPy, Pandas, Matplotlib, Seaborn, and Scikit-Learn are the most widely used Python data science libraries. With their distinct functions and mutual support throughout the data pipeline, these libraries serve as the cornerstone of the data science ecosystem.

In the data science workflow, pre-processing and data cleaning are essential steps that get raw data ready for modelling or analysis. Python libraries that offer effective and user-friendly tools to manage a variety of data preparation tasks, like NumPy, Pandas, and Scikit-Learn, are essential to this process.

A critical phase in data science is exploratory data analysis (EDA), which aids in comprehending the underlying relationships, distributions, and patterns in a dataset prior to modeling. Python libraries like Pandas, Matplotlib, Seaborn, and NumPy are crucial for conducting exploratory data analysis (EDA) because they offer flexible tools for analyzing data and producing statistical and visual insights.

Extracting insights and using data to inform decisions require statistical data analysis. Libraries like NumPy, SciPy, Pandas, Statsmodels, and Scikit-Learn provide a wealth of tools for performing different statistical analyses in Python. Data scientists and statisticians rely on these libraries because they offer functions and methods that make it easier to calculate, interpret, and visualize statistical metrics.

Python machine learning model creation requires the use of a number of frameworks and libraries that offer crucial features and tools for data pre-processing, model construction, testing, and deployment.

There are many features to using Python libraries for data science, which makes it a desirable option for scientists, engineers, and data analysts.

- Because Python libraries are frequently modular, users can import just the parts they require for their analysis. This modularity lowers the memory footprint and aids in dependency management.
- Data manipulation (Pandas), numerical computing (NumPy), visualization (Matplotlib, Seaborn), machine learning (Scikit-Learn, TensorFlow, PyTorch), and statistical analysis (Statsmodels) are just a few of the many data science-related libraries available in Python. Because of this wide range, users can locate specialized tools that are appropriate for their particular tasks.
- Python libraries frequently complement one another, enabling users to combine features from several libraries to create intricate workflows. For instance, it is easy to use Matplotlib for visualization, Scikit-Learn for modeling, and Pandas for data manipulation.
- Numerous libraries offer high-level APIs that abstract complicated implementations, allowing users to complete complex tasks more easily without having to comprehend the underlying algorithms. This is particularly true for deep learning libraries like Keras.
- Robust data structures (such as DataFrames and ndarrays) that support a variety of data formats and enable effective data handling, manipulation, and analysis are offered by libraries like Pandas and NumPy.
- Large-scale data visualization capabilities are offered by libraries such as Matplotlib, Seaborn, and Plotly, which enable users to produce a variety of static, animated, and interactive plots to better understand their data.
- The majority of Python libraries come with thorough documentation and tutorials, which facilitates users' rapid learning and implementation of features.
- Developer and user communities actively support the majority of Python libraries. As a result, there is constant development, regular updates, and an abundance of resources, such as tutorials, documentation, and user support forums.

There are numerous advantages of using libraries in python.

- Python is renowned for its easy-to-understand syntax and libraries' intuitive design, which enable users to pick up and apply data science methods fast and without significant learning curves.
- Libraries' pre-built functions and methods enable data scientists to quickly prototype and develop solutions, speeding up the development process as a whole.
- Numerous libraries are built to effectively manage big datasets. To work with datasets larger than memory, Dask extends Pandas, and PySpark makes use of distributed computing.
- A full data science workflow, from data ingestion to model deployment, is made possible by the combination of libraries, which offer a complete toolkit for conducting statistical analysis, machine learning, and exploratory data analysis (EDA).
- Modern machine learning and deep learning algorithms and techniques are accessible through libraries like Scikit-Learn, TensorFlow, and PyTorch, enabling users to apply sophisticated analytical solutions.
- Data scientists are able to effectively communicate their findings to stakeholders and aid in decision-making by using visualization-specific libraries.
- Python libraries offer a comprehensive approach to data science by integrating easily with other technologies, including web frameworks for application development, cloud platforms for deployment, and SQL databases for data retrieval.
- The majority of Python libraries are open source, meaning that anyone can use, alter, and share them without restriction. In the community, this encourages creativity and cooperation.
- Data ingestion and manipulation are made easier by libraries like Pandas and NumPy, which support a variety of data formats, including CSV, Excel, JSON, and SQL databases.
- Analysis reproducibility is improved by using well-known libraries. Studies can be more successfully replicated by researchers using consistent APIs and methodologies.

Two essential Python libraries for data analysis, scientific computing, and data manipulation are NumPy and Pandas. They offer crucial resources for effectively managing massive volumes of data.

A core Python library for numerical computation is called NumPy. It supports random number generation, mathematical functions, arrays (especially multidimensional arrays), and more.

Key features of NumPy are depicted as follows:

N-Dimensional Arrays: Multi-dimensional arrays and matrices can be supported by NumPy's robust ndarray object.

Broadcasting: Expands smaller arrays to fit the shape of larger arrays by performing element-wise operations on arrays of various shapes. Numerous mathematical functions, including algebraic, statistical, and trigonometric functions, are available.

Random Number Generation: Facilitates random number generation, distribution functions, and random sampling. Functions for linear algebraic operations, such as matrix multiplication, inversion, and decomposition, are available in linear algebra.

Effective Memory Usage: Compared to standard Python lists, this method uses contiguous blocks of memory to enable faster and more effective computations.

Fortran and C/C++ integration: Enables Python to call C, C++, and Fortran code for performance enhancement.

Interoperability: Working smoothly with numerous other libraries, such as SciPy, Pandas, and scikit-learn.

NumPy has several advantages describes as follows:

Speed and Performance: Low-level, optimized C code makes operations faster than lists.

Memory Efficiency: Arrays use less memory and take up less room.

Support for Complex Operations: Arrays can be directly subjected to complex mathematical operations using NumPy.

Compatible with a wide range of other libraries: It compatibles with SciPy, Pandas, and scikit-learn, it operates with ease.

Versatile: Capable of a variety of operations, such as matrix manipulations, slicing, indexing, and reshaping.

Another important library is Pandas. It is built on top of NumPy. Pandas is a high-level data manipulation library. For working with structured data, including databases, spreadsheets, and data tables, it offers functions and data structures.

Following are the main features of Pandas:

Data Frame Object: Labelled, two-dimensional data structures that are simple to work with and examine are made possible by the DataFrame Object.

Series Object: Time series and other linear data can benefit from the use of a series object, which is a one-dimensional array-like object with labelled indices.

Data Manipulation Features: Facilitates data pivoting, reshaping, filtering, grouping, and merging.

Managing Missing Data: Offers strategies for efficiently managing missing data and NaN values.

Data Analysis Functions: Data analysis functions include time-series functionality, statistical operations, filtering, and aggregations.

Connectivity with Other Libraries: works well with other data visualization libraries, such as Seaborn and Matplotlib.

Input/Output: Read and write to a variety of formats, such as JSON, Excel, CSV, and SQL databases.

Pandas has several advantages as follows:

Usability: Makes data loading, cleaning, and preparation procedures simpler.

Data cleaning: Provides tools to deal with duplicate values and missing data with ease.

Data Wrangling: Facilitates effective data manipulation methods such as pivoting, groupby, and merging.

Data Visualization: Seaborn and Matplotlib are compatible with this tool.

Strong Grouping and Aggregation: Makes data aggregation and summarization simple.

IO Capabilities: Reads and writes to a number of file types, including Excel, CSV, SQL, and JSON.

Extremely Scalable: Works well with big datasets and can be made more efficient by using other libraries (like Dask).

One of the most popular Python libraries for data visualization is called Matplotlib. It provides a versatile and strong platform for producing a large range of static, animated, and interactive visualizations. It was created by John D. Hunter.

Because of Matplotlib's adaptable architecture, which is based on NumPy arrays, users can produce both straightforward plots and more intricate graphics. It comes with a Pyplot module that makes plotting easier by offering an interface akin to MATLAB. Because of its many

customization options, Matplotlib is especially helpful for producing figures that are suitable for publication. To produce understandable and perceptive visualizations, it is frequently combined with other data libraries such as Pandas and Seaborn.

Seaborn is another important library for visualization in python. Built on top of Matplotlib, Seaborn is a high-level Python data visualization library. It makes it easier to create statistical plots that are both aesthetically pleasing and educational. Seaborn is especially well-liked because it uses very little code to produce visually appealing visualizations. It is a go-to library for data scientists who want to swiftly analyze and visualize data because it works particularly well with Pandas DataFrames.

Seaborn was created to enhance the visual appeal of standard Matplotlib plots and facilitate the creation of complex visualizations. It offers a user-friendly interface for creating statistical graphics and a wide range of themes and color schemes. Seaborn is perfect for examining patterns and trends in data because it is frequently used to create statistical plots such as regression plots, distribution plots, and heatmaps.

Check your progress-2

(a) Why python is popular language for Data Science?

(b) A core Python library for numerical computation is called _____.

(c) SQL, Julia, SAS, Java and Scala languages are used widely in Data Science.(True/False)

(d) List out the most widely used Python data science libraries.

(e) _____ library of Data Science is popular for data visualization.

4.5 Let us sum up

In this unit we have discussed various applications of Data Science in different popular sectors. The unit also discussed the popular programming languages used in Data Science with advantages. The unit also discussed the libraries and frameworks used for Data Science.

4.6 Check your progress: Possible Answers

1-a True

1-b Data science is used by financial institutions to identify fraudulent transactions and control risks. Banks analyse transaction data to identify suspicious activities and prevent financial crimes. Credit Scoring, Risk Analysis, Algorithmic trading, Fraud Detection Systems are examples of Data Science applications in this field.

1-c Today's era of social media and Data Science play very important role in this field as well. Data science is used by social media companies like Facebook and Netflix to assess user activity and offer tailored content recommendations. Content Recommendation, Sentiment Analysis, Trend Analysis are best examples of Data Science in social media and entertainment.

1-d Data Science plays a vital role in education sector. Data science is used by educational institutions to enhance learning outcomes and customize instruction. Learning platforms make personalized learning path recommendations by analysing student performance data. Best applications of Data Science in education are: Student Performance Analysis, Adaptive Learning Systems, and Course Recommendation etc.

2-a Due to its readability, ease of use, and a wealth of libraries such as Pandas for data manipulation, NumPy for numerical calculations, Matplotlib and Seaborn for visualization, and Scikit-learn for machine learning, Python is the most widely used language in data science.

2-b NumPy

2-c True

2-d NumPy, Pandas, Matplotlib, Seaborn, and Scikit-Learn are the most widely used Python data science libraries

2-e Matplotlib

4.7 Further Reading

1. https://www.geeksforgeeks.org/data-science/major-applications-of-data-science/
2. https://www.datacamp.com/blog/top-programming-languages-for-data-scientists-in-2022

3. https://www.dataquest.io/blog/15-python-libraries-for-data-science/

4.8 Assignments

1. What is the role of Data Science in Healthcare sector?
2. Explain the role of Data Science in Education.
3. What are the applications of Data Science in Business and E-commerce domain?
4. Explain advantages of Python programming language in context to Data Science.
5. Explain advantages of R programming languages.
6. Explain some popular libraries used in Python.

Block-2

Data Collection, Preprocessing, and Management

Unit-1: Data Collection Techniques

Unit Structure

1.0 Learning Objectives

After studying this unit, you should be able to:

- Explain what data collection means and why it is the very first and most important step in any data science project.
- Describe the difference between internal and external data, and between primary and secondary data, with day-to-day examples.
- List and compare different data collection methods such as surveys, interviews, observations, web scraping and APIs.
- Understand basic sampling techniques and decide which one to use in a given situation.
- Identify common challenges that occur during data collection and suggest simple ways to handle them.
- Recognise popular tools used by data professionals to collect data quickly and accurately.

1.1 Introduction to Data Collection

In your daily life, you collect data without even realising it. When you note down attendance in a register, write down marks of a class test, ask your friends which movie they want to watch, or check how many likes a post got on Instagram, you are collecting data. Data collection in the world of computers and data science works in the same way, only on a much bigger scale and with proper planning.

Data collection is the process of gathering and measuring information from different sources so that we can answer questions, prove or disprove an idea, or take a decision. In a data science project, the quality of every step that comes after, such as cleaning, analysis, charts and machine learning, depends on how good the data is. If the data is wrong or missing, even the best computer program in the world cannot give us a correct answer. There is a popular saying among data scientists, garbage in, garbage out, which means if we put bad data into our system, we will get bad results.

Data can come from many places. A few simple examples that a BCA student can relate to are listed below.

- Marks of students in a class, kept in an Excel sheet by the teacher.
- Login records of users on a college website, stored in a database.

- Tweets and posts about a new mobile phone, available on Twitter or Instagram.
- Temperature readings collected automatically by a sensor in a smart classroom.
- Sales records of an online shop like Flipkart or Amazon, stored in their servers.

Data collection can be done in two broad ways.

Manual data collection: Here a person collects the data by hand. For example, a teacher giving a paper survey to students, or a researcher conducting face-to-face interviews.

Automated data collection: Here computer programs or sensors collect data automatically. For example, a Python script that downloads cricket scores from a website every hour, or a fitness band that records the number of steps you walk every day.

Choosing the right method of collection is very important because it affects every later stage of a data science project. This unit explains the main methods used to collect data, the most common sampling techniques, the typical challenges that come up, and the popular tools that make our work easier.

1.1.1 Why is Data Collection Important?

The importance of good data collection can be understood from the following points.

Models are only as good as the data given to them. A weather prediction app trained on wrong temperature readings will give wrong forecasts.

- Good data helps organisations take better decisions, improve their services, and design better products.
- With the right data, hidden trends and patterns can be discovered. For example, an online shopping company can find out which product is most popular in which city.
- Clean, well-collected data reduces the time and effort spent later on cleaning and correcting it.
- Data collected ethically and with permission helps in building trust with customers and users.

1.2.1 Sources of Data: Internal vs External

In any organisation, data may come from inside the organisation itself, or from outside. Both types are useful in different situations. The table below explains the difference using simple examples.

Table 1.1: Internal Data vs External Data

Point	Internal Data	External Data
Where it comes from	Generated inside the organisation from its own work and systems.	Collected from sources outside the organisation, such as websites, public databases, or paid data providers.
Examples	College admission records, library issue records, mess attendance, fee payment data.	Government census data, weather data, Twitter posts, results published on board websites.
Good points	Easily available, well organised, fully under the control of the organisation, easier to keep private.	Provides a much wider view, useful for comparing with competitors, helps in market research.
Difficulties	Limited in scope, may not give the full picture, sometimes stored in different departments separately.	Quality may not be reliable, format may be different, may need a paid subscription, may have legal limits.

1.2.2 Types of Data: Primary and Secondary

When we collect data, we should also think about whether we are collecting it ourselves for the first time, or using data that someone else has already collected. The two types are called primary data and secondary data.

Table 1.2: Primary Data vs Secondary Data

Point	Primary Data	Secondary Data
Meaning	Data that is collected by you (or your team) for the very first time, for a specific purpose.	Data that has already been collected by someone else, and you are using it for your own purpose.

Point	Primary Data	Secondary Data
How it is collected	Surveys, interviews, lab experiments, direct observation, focus group discussions.	Books, research papers, government websites, company reports, public datasets like Kaggle.
Examples	A survey done by you in your college canteen to find favourite snack; readings from a sensor placed by you in the lab.	Census data of India downloaded from the official site; cricket statistics taken from ESPN Cricinfo.
Good points	Exactly fits your purpose, you control the quality, and the data is the latest.	Saves time and money, easy to start with, useful when you need data over many years.
Difficulties	Takes more time and money, needs trained people, may have personal bias.	May not perfectly fit your purpose, may be old, you cannot check how carefully it was collected.

1.3 Data Collection Methods

Different situations call for different methods of data collection. The most common methods used in data science are surveys, interviews, observations, experiments, web scraping and APIs, and social media data. We now look at each one in simple terms with examples that a BCA student can connect with.

1.3.1 Surveys and Questionnaires

A survey is a set of questions given to a group of people to know their opinions, choices, or behaviour. The list of questions is called a questionnaire. Surveys can be conducted online (through Google Forms, Microsoft Forms, SurveyMonkey), over a phone call, in person, or through paper forms.

For example, suppose you want to know which programming language BCA students of your college prefer to learn first. You can create a Google Form with questions like:

- Which year of BCA are you studying in?

- Which programming language do you find easiest? (Choose one: C, Python, Java, JavaScript)
- How many hours per week do you practise coding? (Choose: less than 2, 2-5, 5-10, more than 10)

Most surveys mix two types of questions:

Closed-ended questions: The user must pick from given options. For example, multiple choice or yes/no. Easy to count and analyse.

Open-ended questions: The user can write their answer in their own words. For example, "What feature would you like to add to our college app?" These are richer but harder to analyse.

The four main types of surveys are:

Online Surveys: Forms shared through email, WhatsApp or social media. Tools like Google Forms make this very easy. They are fast, free, and reach many people quickly.

Telephonic Surveys: Conducted over a phone call. Banks and telecom companies use them to ask customers about service quality.

In-Person Surveys: The interviewer meets people face to face. Common in college projects and field studies.

Mail Surveys: Paper forms sent through post. Almost not used today because online forms are faster.

Advantages of surveys are:

- Can reach a very large number of people quickly, especially through online forms.
- Cheap, especially when done online.
- Easy to analyse using simple charts and percentages.

Challenges of surveys are:

- Response bias: people sometimes give answers they think will please the asker, instead of the truth.
- Many people simply ignore online survey links, so the response rate is low.
- Designing good questions is harder than it looks. Confusing or leading questions give wrong results.

1.3.2 Interviews and Focus Groups

An interview is a one-to-one discussion where the interviewer asks questions and the respondent answers them. Unlike a survey, an interview allows follow-up questions and detailed answers.

There are three styles of interviews:

Structured Interview: All questions are decided in advance and asked in the same order to every person. Used in job placement drives where many candidates are interviewed.

Semi-structured Interview: Main questions are decided in advance, but the interviewer can ask extra questions based on the answer. Used by news reporters.

Unstructured Interview: More like an open conversation. Used in research where the topic is new and the researcher wants to explore freely.

A focus group is a small group, usually six to ten people, who discuss a topic together while a moderator guides the conversation. Companies use focus groups to test reactions to a new product or advertisement before launching it.

For example, a mobile company may invite a focus group of college students to discuss what they think of a new phone design. The discussion can bring out ideas that no single survey question would have captured.

Advantages of interviews and focus groups are:

- Give very detailed and rich information.
- The interviewer can ask follow-up questions to dig deeper.
- Personal contact often leads to more honest answers.
- Focus groups bring out new ideas through discussion among participants.

Challenges include:

- Time-consuming and need trained interviewers.
- Can be affected by the personal opinions or behaviour of the interviewer.
- In a focus group, one or two strong-minded participants may dominate the discussion.
- Hard to do for a very large number of people.

1.3.3 Observations and Experiments

Sometimes, the best way to collect data is to simply watch and record what people or systems do. This is called observation. In an experiment, we go one step further and change something on purpose to see what effect it produces.

Observation Examples: Counting how many students enter the college library between 10 am and 12 noon every day; recording how visitors move around in a shopping mall; noting which icon students click first on a college app screen.

Two types of observation are:

Naturalistic Observation: Watching people in their normal environment without disturbing them. For example, observing customer behaviour inside a real shop.

Controlled Observation: Watching people in a controlled setup like a lab. For example, observing how students perform a coding task in the computer lab when given a time limit.

Advantages of observation are:

- Gives real, unbiased information about actual behaviour.
- Useful when people cannot easily explain what they do.

Challenges include:

- Can take a lot of time.
- The observer may interpret things based on personal opinions.
- Privacy must be respected; we cannot observe people without permission.

Experiments involve changing one thing (called a variable) and seeing the effect on something else. Two main types are:

Lab Experiments: Done in a fully controlled environment. For example, testing how fast a sorting algorithm runs on a computer with different input sizes.

Field Experiments: Done in real-world settings with some level of control. For example, testing whether students learn better with online quizzes by trying it in two real classrooms.

Advantages of experiments are:

- They allow us to find cause-and-effect relations, which is the strongest kind of evidence.
- They give very precise data because everything is controlled.

Challenges include:

- They can be expensive and need careful planning.
- Lab results may not always work the same way in real life.

1.3.4 Web Scraping and APIs

Two of the most popular automatic methods used by data scientists to collect data from the internet are web scraping and APIs. Both are very useful for BCA students because they involve writing programs.

Web Scraping

Web scraping is the process of writing a program that automatically reads the content of a web page and pulls out the useful information. For example, you might write a Python program that visits Flipkart, opens the page of a mobile phone, and saves the price into a CSV file. If you do this every day, you can later draw a chart of how the price changed over time.

In Python, two libraries are very commonly used for web scraping:

Requests: Used to download the HTML of a web page.

BeautifulSoup: Used to read the HTML and pick out specific tags such as titles, prices, or links.

A simple scraping program looks like this. Even if you have not learned scraping yet, the structure is easy to follow.

```
import requests
from bs4 import BeautifulSoup
url = 'https://example.com/books'
page = requests.get(url)
soup = BeautifulSoup(page.text, 'html.parser')
# pick out all <h2> tags which contain book titles
for h2 in soup.find_all('h2'):
    print(h2.text)
```

Line by line: we import the libraries, store the address of the page in a variable, download the page, give it to BeautifulSoup, and then loop through every h2 tag printing its text. With small changes, this can collect prices, ratings, news headlines, and many other things.

Advantages of web scraping are:

- Gives access to huge amounts of public data, such as product reviews or stock prices.
- Saves a lot of time, because the program does the work automatically.
- Once written, the program can run again and again without extra effort.

Challenges of web scraping include:

Legal issues: Some websites do not allow scraping. We must always read the website's terms of use and robots.txt file before scraping.

Dynamic content: Many modern websites load data using JavaScript, which makes simple scrapers fail. Tools like Selenium are then needed.

Website changes: If the website changes its design, our program may stop working until we update it.

APIs (Application Programming Interfaces)

An API is a polite, official way for one program to ask data from another program. Many websites offer APIs so that developers do not have to scrape their pages. For example, Twitter, OpenWeather, GitHub, and Google Maps all provide APIs.

Imagine an API as a waiter in a restaurant. You (your program) tell the waiter what you want, the waiter goes to the kitchen (the website's database), and brings back the data in a clean, ready-to-use format, usually JSON.

A simple API call in Python using the requests library looks like this:

```
import requests
city = 'Anand'
api_key = 'your_personal_api_key_here'
url = f'https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}'
response = requests.get(url)
data = response.json()
```

```
print('City:', data['name'])
print('Weather:', data['weather'][0]['description'])
```

Here, we tell the OpenWeather API the name of a city and our personal key, the API returns the data as JSON, and we print just the parts we need. The result is the current weather of Anand, ready to be used in any program or chart.

Advantages of APIs are:

- Provide clean, structured and reliable data, often updated in real time.
- Easier and safer than web scraping because the website itself permits the use.
- Most APIs return data in JSON, which Python can read very easily.

Challenges of APIs include:

- Most APIs put a limit on how many requests you can send per minute or per day. This is called a rate limit.
- You need basic programming knowledge to call an API and read its response.
- Some APIs are paid or require a subscription for full access.

1.3.5 Social Media and Crowdsourced Data

Social media platforms such as Twitter, Instagram, Facebook and YouTube produce a huge amount of data every second. People share opinions, photos, videos, comments, likes and locations. All of this is a goldmine for data scientists who want to understand what users are thinking.

For example, when a new movie releases, companies analyse Twitter posts containing the movie name to find out whether viewers are happy or unhappy. This is called sentiment analysis.

Advantages of social media data are:

- Real-time view of what people are talking about right now.
- Useful for sentiment analysis, brand tracking and customer feedback.
- A very large and diverse pool of users.

Challenges include:

- The data is unstructured and noisy. Posts contain emojis, slang, spelling mistakes, and short forms.

- Privacy and ethics must be respected. We must not collect personal information without consent.
- Users on a particular platform may not represent everyone in the population.

Crowdsourced Data

Crowdsourced data is data that comes from a large number of volunteer contributors. Wikipedia is a famous example. Many people, mostly volunteers, write and edit its articles. OpenStreetMap is another, where people add details of roads and buildings on a free world map. In data science, Kaggle is a popular platform where people share datasets and compete to build the best models. As a BCA student, you can download many free datasets from Kaggle to practise machine learning.

Advantages of crowdsourced data are:

- A very large amount of varied data is available.
- Often kept up to date by the community.
- Encourages open sharing and learning.

Challenges include:

- Quality varies because contributors have different levels of expertise.
- Some data may be incomplete or inconsistent.
- It is hard to control bias when contributors come from one type of background.

Check Your Progress - 1

a) What do you mean by data collection? Give two examples from your daily life.

b) Differentiate between primary data and secondary data with one example each.

c) True or False: Web scraping always gives perfectly clean and structured data.

d) Fill in the blank: A _____ is a polite, official way for one program to ask data from another program.

e) What is crowdsourced data? Give two examples and one challenge of using it.

1.4 Data Sampling Techniques

In many real-world projects, the population of interest is so large that we cannot collect data from every single member. For example, if we want to know the favourite mobile phone of all college students in Gujarat, it is not possible to ask every single student. Instead, we ask a smaller group, called a sample. The way we choose this sample is very important. If the sample is chosen badly, the results will not match the real population.

Sampling techniques are divided into two big groups: probability sampling and non-probability sampling.

1.4.1 Probability Sampling

In probability sampling, every member of the population has a known and equal (or at least non-zero) chance of being selected. This makes the sample more fair and the results more reliable. Common probability sampling methods are explained below.

Simple Random Sampling

Every member of the population has an equal chance of being chosen. Imagine writing the roll number of every BCA student on a small slip of paper, mixing them in a box, and drawing 30 slips with closed eyes. That is simple random sampling.

Advantages: It is very fair and easy to understand.

Challenges: It needs a complete list of the whole population, which may be hard to get.

Stratified Sampling

First, the population is divided into groups, called strata, based on some property such as gender, age, or year of study. Then a random sample is taken from each group.

For example, if you want to survey BCA students about their study habits, you can divide them into First Year, Second Year and Third Year groups, and pick 20 students randomly from each group. This way, every year is properly represented.

Advantages: Very useful when there are clear sub-groups in the population.

Challenges: Needs information about the groups beforehand.

Systematic Sampling: In systematic sampling, we pick every nth member from a list. For example, if you have a list of 1000 customers and you want a sample of 100, you pick every 10th customer (the 10th, 20th, 30th and so on).

Advantages: Easy to use, especially when the list is already in order.

Challenges: If the list has a hidden pattern that matches the picking interval, the sample may be biased.

Cluster Sampling

The population is first divided into clusters, often based on geography. Then a few clusters are chosen at random, and all members of those clusters are surveyed.

For example, to study eating habits of college students in Gujarat, you can treat each college as a cluster, randomly pick five colleges, and survey all the students in those five colleges.

Advantages: Saves time and travel cost, especially when the population is spread out.

Challenges: If clusters are very different from each other, results may not be accurate.

Multistage Sampling

This is a combination of two or more sampling methods. For example, first randomly pick five districts of Gujarat (cluster sampling), then within each district randomly pick three colleges, and finally within each college pick 20 students using simple random sampling. This is useful for very large populations.

Advantages: Highly flexible; works well for big and complex populations.

Challenges: More complex to plan; errors at each stage can add up.

1.4.2 Non-Probability Sampling

In non-probability sampling, every member does not get an equal chance. These methods are quicker but the results may not represent the whole population. They are often used when the proper sampling list is not available, or when we just need a rough idea quickly.

Convenience Sampling

As the name suggests, you simply pick whoever is most convenient or available. For example, asking the first 50 students you meet at the college canteen about their favourite snack.

Advantages: Quick, cheap, and useful for small pilot studies.

Challenges: High risk of bias because the sample may not represent everyone.

Judgmental (Purposive) Sampling

Here, the researcher uses personal judgement to pick people who are believed to be the most informed or representative. For example, choosing five expert teachers in your college for a focus group on online learning.

Advantages: Useful when you need expert opinions or want to study a particular sub-group.

Challenges: Personal bias can affect the choice.

Snowball Sampling

This method is used when the population is hard to reach. The first few participants are asked to suggest more participants, who then suggest others, and so on, like a snowball rolling down a hill and growing in size.

For example, if you want to study free-software contributors in Gujarat, you can start with two or three contributors you know, and ask each of them to refer others.

Advantages: Useful for hard-to-reach or niche groups.

Challenges: The sample tends to be biased because all participants come from one social network.

Table 1.3: Summary of Sampling Techniques

Type	Technique	Best Used When
Probability	Simple Random	Full population list is available and we want fairness
Probability	Stratified	Population has clear sub-groups we must represent
Probability	Systematic	We have an ordered list and want quick selection
Probability	Cluster	Population is spread out across many locations
Probability	Multistage	Population is huge and complex
Non-Probability	Convenience	We need quick, low-cost feedback
Non-Probability	Judgmental	We need expert opinion on a special topic
Non-Probability	Snowball	The target group is hidden or hard to reach

1.5 Challenges in Data Collection

Collecting data sounds simple, but in reality many problems can come up. If we are not careful, the data we collect may give wrong results, no matter how clever our analysis is. The most common challenges are explained below.

1.5.1 Data Quality Issues

Data quality means how accurate, complete, and reliable the data is. Poor-quality data leads to wrong conclusions. Three main types of data quality issues are described below.

A. Inaccurate Data

Data is inaccurate when it does not reflect the real value. For example, a student's age recorded as 200 years, or a phone number with only six digits. Such errors usually come from typing mistakes or wrong measurements.

How to handle it:

- Add data validation rules. For example, age must be between 1 and 120.
- Use automatic tools that flag suspicious values.
- Train people who enter data to follow proper steps.

B. Data Redundancy

Redundancy means the same data is stored more than once. For example, the same student is listed twice in a college database with slightly different spelling of the name. This makes the dataset look bigger than it is and gives wrong counts.

How to handle it:

- Use unique IDs (like Roll Number or Enrolment Number) for every record.
- Run deduplication scripts that find and remove duplicate rows.
- Clean and update the database regularly.

C. Inconsistent Data

Data is inconsistent when the same thing is written in different ways across the dataset. For example, a city written as Ahmedabad in one place, A'bad in another, and AMD in a third. Such inconsistency makes counting and grouping very hard.

How to handle it:

- Set fixed rules for how data should be entered.
- Use lookup tables and standard codes.
- Add consistency checks during preprocessing.

1.5.2 Missing Data

Missing data means some values are not recorded. In a survey, a few people may skip questions. In a dataset, some cells may be empty or marked as NA. Missing data is a very common challenge and you will see it in almost every real dataset.

Missing data can be classified into three types based on why it is missing:

MCAR (Missing Completely at Random): The missing values have no pattern at all. Example: a sensor was switched off for an hour due to power failure, so some readings are missing for that hour.

MAR (Missing at Random): The missingness can be explained by other observed data. Example: older people are less comfortable filling online forms, so age-related answers are more often missing for senior respondents.

NMAR (Not Missing at Random): The missing value depends on the value itself. Example: people earning very high or very low income may refuse to share it. The missingness is hiding the very thing we want to know.

How to handle missing data:

- Imputation: replace missing values with a reasonable estimate, such as the average, median, or most common value.
- Deletion: drop rows or columns that have too many missing values, but only if it does not destroy the dataset.
- Use special algorithms like KNN imputation that fill missing values based on similar rows.

1.5.3 Other Common Challenges

- Privacy and consent: We must not collect personal information without informing the user. The Information Technology (IT) Act in India and the Digital Personal Data Protection Act lay down rules that must be followed.
- Cost and time: Some methods, like face-to-face surveys across many cities, are expensive and slow.

- Language barriers: In a multilingual country like India, the same question may be understood differently in different languages.
- Technical errors: Network failures, hard disk crashes, or software bugs can corrupt data.
- Bias of the collector: Sometimes, the person collecting data unknowingly favours certain answers, which makes the result wrong.

Check Your Progress - 2

1. Fill in the blank: _____ sampling technique ensures that every member of the population has an equal chance of being selected.
2. Which type of missing data is the most dangerous to handle and why? (Hint: think of NMAR.)
3. List any two methods used to handle missing data.
4. Differentiate between probability sampling and non-probability sampling in two lines.
5. Give two simple ways to handle inaccurate data in a college admission database.

1.6 Tools for Data Collection

Many ready-made tools are available that make data collection faster, more accurate, and easier. As a BCA student, you should be familiar with the names and basic uses of the following tools because most of them are widely used in the industry.

1.6.1 Survey and Form Tools

These tools help in creating online forms and analysing the responses.

Table 1.4: Popular Survey Tools

Tool	What it Does
Google Forms	Free, simple form builder. Responses are automatically stored in Google Sheets. Most popular for college projects.
Microsoft Forms	Similar to Google Forms but works inside the Microsoft 365 environment, with results saved in Excel.

Tool	What it Does
SurveyMonkey	Has more advanced features like skip logic and detailed analysis, used by companies for professional research.
Typeform	Known for its attractive, conversational style of asking one question at a time.

1.6.2 Web Analytics Tools

Web analytics tools track how visitors use a website or app, what they click, how long they stay, where they come from, and so on.

Table 1.5: Web Analytics Tools

Tool	What it Does
Google Analytics	Tracks website visitors, page views, time spent, source of traffic, conversions, and much more. Free for most users.
Hotjar	Shows heatmaps where users click most, and records sessions to understand user behaviour.
Mixpanel	Used to track specific user actions inside an app, such as button clicks and feature usage.

1.6.3 Web Scraping Tools and Libraries

These tools and libraries are used to write programs that automatically extract data from web pages.

Table 1.6: Web Scraping Tools

Tool / Library	What it Does
BeautifulSoup	A Python library for parsing HTML and XML. Easy to learn and great for small scraping projects.

Tool / Library	What it Does
Scrapy	A full Python framework for large-scale scraping projects. Supports running many requests in parallel.
Selenium	Controls a real browser. Useful when the website uses JavaScript to load its content.
Octoparse	A no-code scraping tool with a click-and-extract interface, useful for non-programmers.

1.6.4 API Access Tools

These tools and libraries help us call APIs and work with the returned data.

Table 1.7: API Tools

Tool / Library	What it Does
Postman	A graphical app to test APIs without writing code. Very popular among developers.
requests (Python)	The most widely used Python library to call APIs. Easy and powerful.
Twitter API / X API	Provides programmatic access to tweets, user profiles, and trends (now with usage limits).
Google Maps API	Provides map data, location search, and route planning for apps.

1.6.5 Automation and Integration Tools

These tools connect different apps so that data flows from one to another without manual work.

- Zapier: Connects more than 5000 apps. For example, when a new Google Form is filled, automatically add the response to a Slack channel.
- IFTTT (If This Then That): A simpler version of Zapier for personal use. For example, save Instagram photos automatically to Google Drive.
- Apache NiFi: A professional tool used by big organisations to design data pipelines.

1.7 Let us sum up

In this unit, we learned what data collection is and why it is the foundation of every data science project. We discussed the difference between internal and external data, and between primary and secondary data. We saw various data collection methods, including surveys, interviews, observations, experiments, web scraping and APIs, and social media. We also covered probability and non-probability sampling techniques, and learned how to choose the right one. We discussed the most common challenges, such as poor data quality and missing data, and the typical ways to handle them. Finally, we got an overview of popular tools used in the industry, such as Google Forms, Google Analytics, BeautifulSoup, Scrapy, Postman, and Zapier. With these basics, you are now ready to design your own small data collection project, ask the right questions, and choose the right tools and methods.

1.8 Check your progress: Possible Answers

a) Data collection is the process of gathering and measuring information from various sources. Examples: noting attendance in a class register, recording test marks of students.

b) Primary data is collected by the researcher for the very first time (example: survey done in your college). Secondary data is already collected by someone else (example: census data taken from a government website).

c) False. Web scraping data is often messy and needs cleaning.

d) API

e) Crowdsourced data is data collected from a large group of volunteer contributors. Examples: Wikipedia and OpenStreetMap. One challenge is that the quality varies because contributors have different levels of expertise.

f) Simple Random Sampling

g) NMAR (Not Missing at Random) is the most dangerous because the missingness depends on the value itself, which is the very value we want to know. So we cannot estimate it from the rest of the data.

h) Imputation (filling missing values with mean, median, or estimated values) and Deletion (removing rows or columns with missing values).

i) Probability sampling gives every member a known chance of selection, making the sample fair. Non-probability sampling does not, and is used for quick or rough studies.

j) Add data validation rules (for example, age must be between 1 and 120) and use unique IDs like Enrolment Number to avoid duplicate or wrong entries

1.9 Further Readings

1. Doing Data Science, Cathy O'Neil and Rachel Schutt, O'Reilly Media.
2. Web Scraping with Python, Ryan Mitchell, O'Reilly Media.
3. Online articles on data collection at sites such as Simplilearn, Towards Data Science, and Analytics Vidhya.

1.10 Assignments

1. Define data collection. Explain its importance in any data science project with at least three reasons.
2. Differentiate between internal data and external data, and between primary and secondary data, with one example of each from a college environment.
3. Explain any four data collection methods in detail with one advantage and one challenge of each.
4. Compare probability sampling and non-probability sampling. Give one example of each type from a real-world situation.
5. List any five challenges of data collection. For any two of them, suggest practical ways to handle them.
6. Imagine you are a BCA student designing a small project to find out the most preferred online learning platform among college students in your city. Describe step by step how you will collect the data, which method and sampling technique you will use, and which tool you will use to make the form.
7. Write a Python program (using requests and BeautifulSoup) that downloads any public web page and prints the text of all h1 and h2 tags. Show the output for any one website you tried.

Unit-2: Data Preprocessing

Unit Structure

2.0 Learning Objectives

After studying this unit student should be able to:

- After studying this unit, you should be able to:
- Understand what data preprocessing is and why it is one of the most important steps in any data science project.
- Explain the typical steps in a preprocessing pipeline: cleaning, transformation, and reduction.
- Handle missing data using simple techniques like mean, median and mode imputation, and deletion.
- Detect and handle outliers using simple statistical methods such as Z-score and IQR.
- Find and remove duplicate rows from a dataset using Pandas.
- Apply data transformation techniques such as scaling, normalisation and encoding of categorical variables.
- Use simple data reduction techniques such as feature selection and binning to make data easier to work with.

2.1 Introduction to Data Preprocessing

Imagine you have downloaded a dataset of cricket match scores from the internet. You open it in Excel and find that some rows are duplicated, several player names are misspelled, the column for runs has empty cells in many places, and the dates are written in three different formats. If you try to do any analysis on this data straight away, your charts will be wrong and your conclusions will be misleading.

This is where data preprocessing comes in. Data preprocessing is the process of converting raw, messy data into a clean, organised and consistent form so that it is ready for analysis or for training a machine learning model. In simple words, preprocessing is the cleaning and tidying-up step before the real work begins.

Data scientists usually say that preprocessing takes about 70 to 80 per cent of the total time in a project. The actual machine learning model takes only 10 to 20 per cent of the time. This shows how important and time-consuming this step is.

Why is Data Preprocessing Important?

- It improves data quality, which leads directly to better analysis results.

- It removes inconsistencies that can confuse algorithms.
- It helps machine learning models learn faster and more accurately.
- It reduces computational complexity by removing useless or duplicate information.
- It makes the data easier for humans to understand and explore.

Real-Life Examples of the Need for Preprocessing

Online shopping: If a customer enters Mumbai, MUMBAI, mumbai, and Bombay in different orders, all four are the same city. Without preprocessing, the system will think they are four different places.

Banking: A column for income may have some empty cells because customers did not declare income. We need to handle these missing values before predicting loan eligibility.

Health records: A field for blood pressure may have a value like 999, which is clearly wrong. Such values, called outliers, must be detected and corrected.

Why is Data Preprocessing Important?

- Ensures data quality and consistency.
- Helps improve the accuracy and efficiency of data-driven models.
- Reduces computational complexity.

Importance of Preprocessing in Data Science

- Improving Data Quality: Addresses common data issues like noise, missing values, and inconsistencies. Leads to cleaner datasets that can improve machine learning model performance.
- Enhancing Model Accuracy: Techniques such as scaling, encoding, and handling outliers help ensure that data is correctly interpreted by algorithms. Increases the chances of producing more reliable and generalizable models.
- Reducing Computational Complexity: Data reduction techniques can minimize dataset size, improving processing speed and reducing memory requirements. Helps eliminate irrelevant or redundant information that may affect model efficiency.

2.2 The Data Preprocessing Pipeline

A pipeline is a series of steps that data goes through, one after the other. The data preprocessing pipeline has three main stages, as shown below.

Table 2.1: Three Stages of the Preprocessing Pipeline

Stage	What Happens	Common Tasks
Step 1: Data Cleaning	We find and fix errors, missing values, and duplicates in the data.	Handle missing values, remove duplicates, fix outliers and wrong formats.
Step 2: Data Transformation	We change the form of the data so that it suits our analysis or our model.	Scaling, normalisation, encoding categorical variables into numbers.
Step 3: Data Reduction	We make the data smaller without losing important information.	Feature selection, dimensionality reduction (PCA), binning.

In real life, these steps are not always done in a strict order. Sometimes we do a bit of cleaning, then some transformation, find a new problem, and go back to cleaning again. The pipeline is just a guideline; the actual order depends on the project.

2.3 Handling Missing Data

Missing data means that some cells in the dataset are empty or marked with special values like NA, NaN, null, or even -999. This is one of the most common problems in any real dataset.

For example, in a Pandas DataFrame loaded from a CSV file, missing values usually appear as NaN, which stands for Not a Number.

2.3.1 Causes of Missing Data

Before we decide how to handle missing data, it helps to understand why it happened. Some common reasons are:

Data Collection Issues: Surveys may have unanswered questions because respondents skipped them. Sensors may fail because of low battery or network problems. Web scraping programs may miss data when the site structure changes.

Data Storage Errors: Database files may get corrupted, or some records may be lost during migration from one system to another.

Human Error: While typing data manually, people may forget to fill some columns. Sometimes they enter a value in the wrong column.

2.3.2 Types of Missing Data

Statisticians divide missing data into three types based on the pattern of missingness. Knowing the type helps us choose the right method to handle it.

MCAR (Missing Completely at Random): The fact that data is missing has no relation to any other variable. Example: a sensor was switched off for an hour due to power failure, so some readings during that hour are missing.

MAR (Missing at Random): The missingness is related to other observed data. Example: in a survey, female respondents may be more comfortable answering some questions than others. The missingness depends on gender, which we do know.

NMAR (Not Missing at Random): The missingness depends on the value itself. Example: people with very high or very low income may refuse to share it. This is the hardest type to handle, because we cannot guess the missing value from the rest of the data.

2.3.3 Methods for Handling Missing Data

There are two main approaches: imputation (filling in the missing values) and deletion (removing rows or columns).

Imputation Techniques

Mean Imputation: Replace missing numerical values with the mean (average) of that column. Useful when the data is roughly symmetric and has no big outliers.

Median Imputation: Replace missing values with the median (middle value). Better than mean when the data has outliers, because the median is not affected by extreme values.

Mode Imputation: For categorical data (like city or gender), replace missing values with the most common value, called the mode.

KNN Imputation (K-Nearest Neighbors): For each row with a missing value, find the k most similar rows and use the average of their values to fill in the missing one. More accurate but slower.

Forward / Backward Fill: Used for time series data. Forward fill copies the previous value into the missing cell. Backward fill copies the next value.

Deletion Techniques

Listwise Deletion: Drop the entire row if any column has a missing value. Easy and quick, but if many rows have missing values, we lose a lot of data.

Pairwise Deletion: Use whatever data is available for each calculation. For example, when calculating correlation between two columns, only use rows where both have values.

Column Deletion: If a whole column has too many missing values (say more than 50 per cent), it is often better to drop the column entirely.

Table 2.2: Comparing Methods of Handling Missing Data

Method	When to Use	Drawback
Mean Imputation	Numeric data, no big outliers	Reduces variance, may introduce bias
Median Imputation	Numeric data with outliers	Still ignores relationships between columns
Mode Imputation	Categorical data	May make a category falsely common
KNN Imputation	When relationships between columns matter	Slower for large datasets
Listwise Deletion	Only a few missing rows	May lose a lot of useful data
Column Deletion	A column has mostly missing values	We lose a feature completely

2.3.4 Hands-On with Pandas

Pandas provides simple methods to handle missing values. The main ones are isna, fillna and dropna. Let us see them through a small example.

```
import pandas as pd
import numpy as np


# Create a small dataset of student marks
```

```
data = {
   'Name':  ['Asha', 'Bhavin', 'Chirag', 'Disha', 'Esha'],
   'Maths': [78, np.nan, 65, 88, np.nan],
   'Science': [85, 72, np.nan, 90, 80]
}
df = pd.DataFrame(data)
print(df)
```

This creates a small DataFrame with five students and two subjects. Some marks are missing (shown as NaN). Now let us count the missing values in each column.

```
print(df.isna().sum())
```

The isna method returns True wherever there is a missing value, and the sum gives the count column-wise. The output will show that Maths has 2 missing values and Science has 1.

To fill missing values with the mean of the column:

```
df['Maths'] = df['Maths'].fillna(df['Maths'].mean())
df['Science'] = df['Science'].fillna(df['Science'].mean())
print(df)
```

To drop any row that has at least one missing value:

```
df_cleaned = df.dropna()
```

These two methods, fillna and dropna, are the most common ways to handle missing values in Pandas.

Check Your Progress - 1

a) True or False: Data preprocessing is needed only when the dataset is very large.

b) Fill in the blank: The most common technique to fill missing numerical data with the average of a column is called _____ imputation.

c) What is the main goal of data preprocessing?

d) List the three main steps of the data preprocessing pipeline.

e) Give one example each of MCAR, MAR and NMAR missing data.

2.4 Detecting and Handling Outliers

An outlier is a data point that is very different from the rest of the data. For example, in a dataset of marks where most students have scored between 50 and 90, a value of 5 (or 200) would be an outlier.

Outliers can be true unusual values (a top-scoring student getting 99) or errors (someone entered the marks as 990 by mistake). The trick is to find them and decide whether to keep, fix, or remove them.

2.4.1 Why Outliers Matter

- They distort statistical measures. The mean (average) gets pulled up or down by extreme values, which can give a wrong picture.
- They affect machine learning models. Algorithms like linear regression are very sensitive to outliers.
- They may indicate something important. In banking, a very high transaction may not be an error, but a fraud worth investigating.
- They make charts misleading by stretching the axis.

2.4.2 Methods to Detect Outliers

There are two broad ways to find outliers: statistical methods and visual methods.

Statistical Methods

Z-score Method: The Z-score tells us how many standard deviations a value is away from the mean. The formula is:

```
Z = (x - mean) / standard_deviation
```

If the absolute Z-score of a value is greater than 3, the value is generally considered an outlier.

Example: In a dataset of test scores with mean 75 and standard deviation 10, a score of 110 has a Z-score of (110 - 75) / 10 = 3.5, which is an outlier.

Interquartile Range (IQR) Method: The IQR is the spread of the middle 50 per cent of the data. It is the difference between the third quartile (Q3) and the first quartile (Q1).

```
IQR = Q3 - Q1

Lower bound = Q1 - 1.5 * IQR

Upper bound = Q3 + 1.5 * IQR
```

Any value below the lower bound or above the upper bound is considered an outlier.

Example: If Q1 is 40, Q3 is 60, then IQR is 20. The lower bound is 40 - 30 = 10 and the upper bound is 60 + 30 = 90. So any value below 10 or above 90 is an outlier.

Visualisation Techniques

Box Plot: A box plot is a chart that shows the median, the quartiles, and any outliers. The box represents the middle 50 per cent of the data (between Q1 and Q3). Whiskers extend from the box to the smallest and largest non-outlier values. Points beyond the whiskers are drawn as individual dots and are the suspected outliers.

Scatter Plot: A scatter plot shows individual data points on two axes. Points that lie far from the main cluster are likely outliers.

Histogram: A histogram shows how many values fall into each range. A small isolated bar far from the main cluster suggests outliers.

2.4.3 Hands-On Outlier Detection in Python

The following Python program creates a small dataset of student marks, calculates the IQR, and prints the outliers.

```
import pandas as pd
marks = pd.Series([55, 60, 62, 65, 68, 70, 72, 75, 78, 99, 5])
Q1 = marks.quantile(0.25)
Q3 = marks.quantile(0.75)
IQR = Q3 - Q1
lower = Q1 - 1.5 * IQR
upper = Q3 + 1.5 * IQR
outliers = marks[(marks < lower) | (marks > upper)]
print('Outliers:', outliers.tolist())
```

Step by step: we create a Pandas Series of marks, find the first and third quartiles, calculate the IQR, decide the lower and upper limits, and finally pick out values that fall outside the limits. The expected outliers are 5 and possibly 99 depending on the data.

2.4.4 Methods to Handle Outliers

Once we have detected the outliers, we have to decide what to do with them. There is no single correct answer; the choice depends on the situation.

Removal: If the outlier is clearly an error, simply delete that row. Example: a student's age recorded as 200 years is obviously a typing mistake.

Capping (Winsorisation): Replace the outlier with the maximum acceptable value. For example, all incomes above 50 lakh per year may be capped at 50 lakh.

Imputation: Replace the outlier with the mean or median of the column, like we do for missing values.

Transformation: Apply a mathematical transformation such as log to reduce the effect of large values. The log transformation pulls extreme values closer to the rest.

Use Robust Statistics: Instead of the mean, use the median. Instead of the standard deviation, use the IQR. These are not affected much by outliers.

Keep Them: If the outliers are real and meaningful (like fraud transactions or rare diseases), keep them and study them carefully.

2.5 Removing Duplicates

Duplicate data means the same record appears more than once in the dataset. This is a very common problem when data comes from multiple sources or is entered by hand. Duplicates make our counts and averages wrong, and they can also confuse machine learning models.

2.5.1 Common Causes of Duplicates

- Data merging: When two or more datasets are joined, the same records may appear from different sources. For example, customer records from the sales and the marketing department may overlap.
- Manual entry errors: A clerk may enter the same patient record twice without realising it.
- Batch processing problems: An automatic system may submit the same data twice if there is a network glitch.
- Forms submitted multiple times: A user may click the submit button on a Google Form twice.

2.5.2 Methods to Identify Duplicates

- Unique Identifiers: Use a unique field like Roll Number, Aadhaar Number, Employee ID or Order ID to find duplicates.
- Exact Match: Compare two rows on every column. If all columns are equal, they are exact duplicates.
- Statistical or Approximate Match: Sometimes the same person is recorded as Asha and ASHA. Such near-duplicates need fuzzy matching tools.
- Comparison of Selected Fields: Match on key fields like name and date of birth, even if other fields differ.

2.5.3 Methods for Removing Duplicates

Drop Identical Rows: If two rows are identical in every column, keep just one and drop the rest. This is the simplest method.

Merge Partial Duplicates: If two rows refer to the same person but contain different information, merge them so no information is lost. For example, two rows for the same customer with different phone numbers can be merged into one row with both phone numbers.

Keep the Most Recent or Most Complete: If a record exists in two versions, keep the more recent or the more complete one.

Flag for Manual Review: Sometimes it is hard to be sure whether two records are duplicates. In sensitive data like medical records, it is safer to flag them and let a person check.

2.5.4 Removing Duplicates in Pandas

Pandas provides two simple methods, duplicated and drop_duplicates.

```
import pandas as pd
data = {
   'Name': ['Asha', 'Bhavin', 'Asha', 'Chirag', 'Bhavin'],
   'City': ['Anand', 'Surat', 'Anand', 'Rajkot', 'Surat'],
   'Marks': [78, 82, 78, 65, 82]
}
df = pd.DataFrame(data)
# Show which rows are duplicates
```

```
print(df.duplicated())
# Remove duplicate rows
df_clean = df.drop_duplicates()
print(df_clean)
```

The duplicated method returns True for every row that is a duplicate of an earlier row. The drop_duplicates method removes them, keeping only the first occurrence by default. We can also drop duplicates based on selected columns:

```
# Drop duplicates only based on the Name column
df.drop_duplicates(subset=['Name'])
```

2.6 Data Transformation

After cleaning, the next step is to transform the data so that it is in the right form for analysis or for a machine learning algorithm. Data transformation includes activities like scaling numbers, encoding categories into numbers, and applying mathematical functions.

2.6.1 Scaling and Normalisation

Different columns in a dataset often have very different ranges. For example, age may go from 18 to 70, while salary may go from 10,000 to 10,00,000. If we feed such data directly to certain algorithms, the column with the bigger range will dominate, which is not fair. Scaling and normalisation bring all columns to a similar range.

Min-Max Scaling

Min-Max scaling rescales values into a fixed range, usually 0 to 1. The formula is:

```
x_new = (x - min) / (max - min)
```

Example: If marks range from 30 to 90, a mark of 60 becomes (60 - 30) / (90 - 30) = 0.5.

Min-Max scaling is useful in neural networks and image processing where pixel values are often scaled to the range 0 to 1.

Standardization (Z-score Normalisation)

Standardization centres the data so that the mean becomes 0 and the standard deviation becomes 1. The formula is:

```
x_new = (x - mean) / standard_deviation
```

Standardization is useful for algorithms like linear regression, logistic regression, and Support Vector Machines (SVM), which assume that data is roughly centred around 0.

Robust Scaling

Robust scaling uses the median and the IQR instead of the mean and the standard deviation. It is therefore not affected by outliers, and is the right choice when the data has many extreme values.

Table 2.3: Comparing Scaling Methods

Technique	Range	Best Used When
Min-Max Scaling	0 to 1	Neural networks, image data, KNN
Standardization	Mean 0, SD 1	Linear regression, logistic regression, SVM
Robust Scaling	Around the median	Data with many outliers

Hands-On with scikit-learn

The scikit-learn library has ready-made tools for scaling. The two most common ones are MinMaxScaler and StandardScaler.

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler
data = {
   'Age':    [20, 22, 25, 30, 45],
   'Salary': [25000, 30000, 50000, 70000, 120000]
}
df = pd.DataFrame(data)

# Min-Max scaling
mm = MinMaxScaler()
df_mm = pd.DataFrame(mm.fit_transform(df), columns=df.columns)
print(df_mm)
# Standardization
```

```
sd = StandardScaler()
df_sd = pd.DataFrame(sd.fit_transform(df), columns=df.columns)
print(df_sd)
```

After Min-Max scaling, all values lie between 0 and 1. After Standardization, the mean of every column is 0 and the standard deviation is 1. Now Age and Salary are on the same scale and a fair comparison can be made.

2.6.2 Encoding Categorical Variables

Most machine learning algorithms work only with numbers. So if a dataset has categorical columns like Gender (Male, Female), City (Anand, Surat, Rajkot) or Education (High School, Bachelor's, Master's), we must convert these into numbers. The two most common ways are Label Encoding and One-Hot Encoding.

Label Encoding

Label encoding assigns a unique integer to every category.

Example: Education with values High School, Bachelor's, Master's becomes 0, 1, 2.

Label encoding is fine for ordinal categories (where there is a natural order, such as Low, Medium, High). But for plain categories like City, label encoding can confuse the model into thinking that one city is greater than another.

One-Hot Encoding

One-hot encoding creates a new binary column for every category. A row gets the value 1 in the column that matches its category, and 0 elsewhere.

Example: A column City with values Anand, Surat, Rajkot becomes three new columns: City_Anand, City_Surat, City_Rajkot. A row for a student from Anand becomes 1, 0, 0.

One-hot encoding is the safer default for non-ordinal categories. The only drawback is that it creates many new columns when there are many categories.

Hands-On with Pandas and scikit-learn

```
import pandas as pd
data = {
   'Name': ['Asha', 'Bhavin', 'Chirag'],
   'City': ['Anand', 'Surat', 'Anand'],
```

```
    'Education': ['HS', 'BCA', 'MCA']
}
df = pd.DataFrame(data)
# One-hot encoding using Pandas
df_encoded = pd.get_dummies(df, columns=['City'])
print(df_encoded)
```

Pandas adds new columns City_Anand and City_Surat. Now the data is fully numeric and can be passed to a machine learning model.

Check Your Progress - 2

f) Which of the following is NOT a method to detect outliers? (a) Z-score (b) Box plot (c) Mean imputation (d) IQR

g) Fill in the blank: The technique that brings all values into a fixed range like 0 to 1 is called _____ scaling.

h) What is one-hot encoding? Give one example.

i) Why is the median preferred over the mean in robust scaling?

j) Name the Pandas function used to remove duplicate rows.

2.7 Data Reduction

Data reduction means making the dataset smaller without losing important information. This sounds counter-intuitive at first, but think about it like packing for a trip. You take only what you need, not everything you own. Data reduction does the same: keeps the important and removes the rest.

Why Reduce Data?

- Smaller data is faster to process.
- Smaller data uses less memory and storage.
- Removing irrelevant data improves model accuracy.
- Charts become clearer when there are fewer columns.
- It reduces the risk of overfitting in machine learning models.

2.7.1 Feature Selection

Feature selection means picking only the columns (features) that really matter for our analysis. For example, in a dataset for predicting house prices, the area and the number of bedrooms are very important, but the colour of the front door is probably not. Feature selection drops the unimportant columns.

Three common ways to do feature selection are:

- Correlation analysis: Drop features that are not correlated with the target. Also drop features that are very similar to each other (highly correlated with each other).
- Chi-square test: A statistical test for categorical features.
- Feature importance from trees: Algorithms like Random Forest can tell us which features they used the most. We can keep just those.

2.7.2 Dimensionality Reduction with PCA

Sometimes we have very many columns, like 1000 features in an image dataset. Dimensionality reduction is a way to combine them into a smaller number of new features that still capture most of the information.

Principal Component Analysis (PCA) is the most popular method. PCA finds new combined columns, called principal components, that explain the most variation in the data. Often the first 10 components can capture 90 per cent of the information from the original 100 columns.

As a BCA student, you do not need to know the heavy maths of PCA. You just need to know what it does and that scikit-learn provides a ready-made PCA tool.

```
from sklearn.decomposition import PCA
# Suppose X is a numpy array with many columns
pca = PCA(n_components=3)
X_reduced = pca.fit_transform(X)
print('Original shape:', X.shape)
print('Reduced shape:', X_reduced.shape)
```

2.7.3 Removing Redundant Features

Two features are redundant if they carry the same information. For example, height in cm and height in inches are giving us the same thing in different units. We should keep only one. Other

examples are temperature in Celsius and temperature in Fahrenheit, or total marks and percentage from the same total.

Redundant features can be found using correlation. If two features have a correlation close to 1 (or -1), one of them can usually be dropped.

2.8 Data Discretization and Binning

Discretisation is the process of converting a continuous variable into a categorical one by grouping values into bins (intervals). For example, if we have the age of every customer, we can put them into bins like Child (0-12), Teenager (13-19), Adult (20-59) and Senior (60 and above). Binning makes the data easier to read and chart, and it sometimes helps machine learning algorithms that expect categorical data.

2.8.1 Binning Techniques

Equal Width Binning: The total range is divided into intervals of equal width. If the data ranges from 0 to 100 and we want 4 bins, the bins are 0-25, 26-50, 51-75, 76-100.

Equal Frequency Binning: Each bin contains roughly the same number of data points. If we have 100 students and want 4 bins, every bin will have around 25 students. The widths of the bins will not be equal.

Custom Binning: We define the boundaries based on domain knowledge. For age, we may use Child (0-12), Teen (13-19) and Adult (20+).

Table 2.4: Comparing Binning Techniques

Technique	How Bins are Made	Best Used When
Equal Width	Each bin has the same range of values.	When data is roughly evenly distributed.
Equal Frequency	Each bin has the same number of points.	When the data is skewed and we want balanced bins.
Custom	Bins decided manually based on logic.	When we have meaningful boundaries from the domain.

2.8.2 Binning in Pandas

Pandas provides two functions: cut for equal-width binning and qcut for equal-frequency binning.

```
import pandas as pd
ages = pd.Series([5, 12, 17, 22, 28, 35, 50, 62, 70])
# Custom bins with labels
bins = [0, 12, 19, 59, 100]
labels = ['Child', 'Teen', 'Adult', 'Senior']
groups = pd.cut(ages, bins=bins, labels=labels)
print(groups)
# Equal frequency bins
groups2 = pd.qcut(ages, q=3, labels=['Low', 'Mid', 'High'])
print(groups2)
```

In the first part, every age is converted into a category. The second part shows equal-frequency binning, where the cut-off points are chosen so that each bin has the same number of values.

Advantages of Binning

- Makes data easier to interpret. Saying 25 customers are in the Adult group is clearer than listing every age.
- Reduces the effect of outliers. A very high value just goes to the top bin.
- Makes some algorithms more stable, especially decision trees.
- Simplifies graphs and tables for non-technical audiences.

Check Your Progress - 3

k) Fill in the blank: Duplicate data can arise from data merging, manual data entry errors, and _____.

l) Fill in the blank: _____ binning divides the range of data into intervals of equal size.

m) What is the main purpose of data transformation?

n) Name any two ways data reduction helps a machine learning project.

o) What is PCA used for in one short sentence?

2.9 Let us sum up

This unit took you through every important step of data preprocessing. We started with the idea that real-world data is always messy and that preprocessing turns it into a clean form ready for analysis. We learned how to handle missing values using imputation and deletion, and saw how to do this in Pandas using fillna and dropna. We discussed outliers, how to detect them with Z-score, IQR and box plots, and how to handle them by removing, capping, or transforming. We covered the removal of duplicate rows using drop_duplicates. We then moved to data transformation, where we learned scaling techniques (Min-Max, Standardization, and Robust scaling) and encoding techniques (Label Encoding and One-Hot Encoding) with simple Python examples. Finally, we looked at data reduction (feature selection and PCA) and binning. With these skills, you can now take any messy CSV file and turn it into a clean dataset ready for modelling or visualisation.

2.10 Check your progress: Possible Answers

a) False. Data preprocessing is needed for almost every dataset, big or small.

b) Mean

c) The main goal of data preprocessing is to convert raw, messy data into a clean, organised, and consistent form so that it is ready for analysis or for training a machine learning model.

d) The three main steps are Data Cleaning, Data Transformation, and Data Reduction.

e) MCAR: a sensor was off for an hour due to power failure. MAR: older respondents skipped a particular question because they were uncomfortable with online forms. NMAR: very high-income people refused to share their income.

f) (c) Mean imputation. It is used for handling missing data, not for detecting outliers.

g) Min-Max

h) One-hot encoding creates a new binary column for every category. A row gets 1 in the column matching its category and 0 elsewhere. Example: a City column with Anand, Surat, Rajkot becomes three new columns City_Anand, City_Surat, City_Rajkot.

i) Because the median is not affected by extreme values (outliers), while the mean is.

j) drop_duplicates

k) Batch processing issues

l) Equal width

m) The main purpose of data transformation is to change the form of the data so that it suits the chosen algorithm and gives better results, for example by scaling or encoding.

n) Two ways: it makes training faster by reducing data size, and it improves accuracy by removing irrelevant or noisy features.

o) PCA is used to reduce the number of features in a dataset by combining them into fewer new features, while preserving most of the information.

2.11 Further Readings

1. Hands-On Data Preprocessing in Python, Roy Jafari, Packt Publishing.
2. Python for Data Analysis, Wes McKinney, O'Reilly Media.
3. Online tutorials on Pandas at the official Pandas website (pandas.pydata.org).
4. Online tutorials on scikit-learn at scikit-learn.org

2.12 Assignments

1. Explain the importance of data preprocessing with at least three real-life examples from a college environment.
2. Discuss the three types of missing data with one example each. For any two types, describe a suitable method to handle them.
3. Compare and contrast imputation and deletion as methods of handling missing data. When would you choose one over the other?
4. Explain Z-score and IQR methods of detecting outliers, with a small numerical example for each.

Unit-3: Feature Engineering

Unit Structure

3.0 Learning Objectives

After studying this unit student should be able to:

- After studying this unit, you should be able to:
- Define feature engineering and explain why it is one of the most important skills in data science.
- Create new features from existing data using mathematical transformations and interaction features.
- Extract useful features from date, time, and location (geospatial) data.
- Process text data using techniques like tokenization, stemming, lemmatization, TF-IDF, and word embeddings.
- Handle time series data using simple feature engineering ideas like lag features and rolling means.
- Apply feature selection methods such as filter, wrapper and embedded methods.
- Use feature scaling and normalisation correctly for different machine learning algorithms.

3.1 Introduction to Feature Engineering

Feature engineering is the process of creating, modifying, or transforming features (the columns in our dataset) so that machine learning models can learn from them better. In simple words, feature engineering is about giving the model the right inputs.

Imagine you are training a model to predict whether a student will pass an exam. Your raw dataset has columns like Date_of_Birth and Date_of_Exam. By themselves, these dates are not very useful. But if you create a new column Age_in_Years from them, suddenly your model has a much more meaningful feature. That is feature engineering in action.

Many experienced data scientists say that good feature engineering matters more than the choice of algorithm. A simple algorithm with great features often beats a complex algorithm with poor features.

Why Feature Engineering Matters

- Well-designed features can dramatically improve model accuracy.
- Carefully chosen features make a model easier to understand. We can clearly see what the model is using to make decisions.

- Good features reduce overfitting, where a model memorises the training data instead of learning general patterns.
- Feature engineering can simplify the model by removing noise and useless data.
- It allows us to bring our domain knowledge into the model. For example, in cricket, the run rate is a more useful feature than just runs and balls separately.

A Simple Example to Begin With

Suppose we have a dataset of online food orders with these columns: Order_ID, Customer_ID, Order_DateTime, Total_Amount, Number_of_Items. As a data scientist, we can create many useful features from these:

- From Order_DateTime, we can extract Hour_of_Day, Day_of_Week, Is_Weekend, Month.
- From Total_Amount and Number_of_Items, we can create Price_per_Item.
- From Customer_ID and a history of orders, we can create Average_Order_Value or Days_Since_Last_Order.

All these new features were created from the existing data. The model never saw them in the original CSV file but they make prediction much easier.

3.2 Feature Creation

Feature creation is the heart of feature engineering. It is the art of building new, useful columns out of existing ones. Below we look at the most common ways to create new features, with examples that BCA students can relate to.

3.2.1 Mathematical Transformations

Mathematical transformations apply simple maths to a column to bring out hidden patterns or to fix problems like skewness.

Table 3.1: Common Mathematical Transformations

Transformation	What It Does	Example
Logarithmic (log)	Reduces skewness by squashing very large values closer to small ones.	Salary data is often very skewed; applying log makes the distribution more normal.

Transformation	What It Does	Example
Square Root	Mildly reduces the effect of large values; gentler than log.	Used for count data like number of website visits per day.
Polynomial (x squared, x cubed)	Captures non-linear relationships between feature and target.	If house price grows faster than area, x squared (area squared) helps the model.
Reciprocal (1 divided by x)	Inverts values; useful when smaller values have bigger effect.	Distance to bus stop: a small distance has a big effect on choice of route.

Hands-On with Pandas and NumPy

Let us see how to apply a log transformation to a Pandas column.

```
import pandas as pd
import numpy as np
df = pd.DataFrame({
    'Salary': [25000, 30000, 50000, 80000, 250000, 800000]
})
# Apply log transformation. Use np.log1p to handle zero safely.
df['Log_Salary'] = np.log1p(df['Salary'])
print(df)
```

After the log transform, the very large values shrink and become comparable to small ones. This often improves model performance for skewed data.

3.2.2 Interaction Features

Interaction features combine two or more existing features to capture how they work together. Often, the combination is more useful than each feature alone.

Table 3.2: Types of Interaction Features

Interaction Type	Description	Example
Multiplication	Multiply two features.	Price multiplied by Quantity gives Total_Sale.

Interaction Type	Description	Example
Division	Divide one feature by another to get a ratio.	Marks divided by Total gives Percentage.
Addition or Subtraction	Add or subtract features.	Study_Hours plus Class_Hours gives Total_Learning_Hours.
Polynomial Interactions	Multiply a feature by itself or by another (like x times y, or x squared).	Age multiplied by Age (Age squared) for age-related health risks.

Example in code:

```
import pandas as pd
df = pd.DataFrame({
    'Price':    [100, 200, 50, 300],
    'Quantity': [2, 3, 5, 1]
})
# Multiplication interaction
df['Total_Sale'] = df['Price'] * df['Quantity']
# Division interaction
df['Avg_Price_per_Item'] = df['Total_Sale'] / df['Quantity']
print(df)
```

With just two new lines of code, we have created two new and very useful features.

3.2.3 Handling Date and Time Features

Date and time columns are very common but they are not directly usable by most algorithms. We must extract simpler numeric features from them. Here are the most common ones.

Table 3.3: Common Date and Time Features

Feature	Description	Example Use
Day, Month, Year	Break the date into separate parts.	Find which months have highest sales.
Day of the Week	Returns Monday to Sunday (or 0 to 6).	Check whether app usage is higher on weekends.
Is_Weekend	True for Saturday and Sunday, False otherwise.	Predict food order spikes on weekends.
Hour of the Day	0 to 23, useful for time-of-day patterns.	Cab booking demand by hour.
Days Since an Event	Number of days from a fixed date.	Days since a customer's last purchase.

Hands-On Date Feature Creation in Pandas

```
import pandas as pd
df = pd.DataFrame({
   'Order_Date': ['2024-01-05', '2024-01-06', '2024-01-07', '2024-02-15']
})
# Convert string to datetime
df['Order_Date'] = pd.to_datetime(df['Order_Date'])
# Extract useful parts
df['Year']       = df['Order_Date'].dt.year
df['Month']      = df['Order_Date'].dt.month
df['Day']        = df['Order_Date'].dt.day
df['DayName']    = df['Order_Date'].dt.day_name()
df['Is_Weekend'] = df['Order_Date'].dt.weekday >= 5
print(df)
```

With just six lines, we converted a single date column into five new useful features. The dt accessor in Pandas is your best friend for date and time work.

3.2.4 Geospatial Features

Geospatial data refers to information that has a location, like latitude and longitude. From these raw coordinates, we can create many meaningful features.

Table 3.4: Common Geospatial Features

Feature	Description	Example
Latitude and Longitude	Numeric coordinates of a location.	Coordinates of a delivery address.
Distance to a Point	Distance from one location to another important point.	Distance to the nearest hospital, mall, or metro station.
Region or Pin Code	Group locations by area.	Sales by state or by pin code.
Spatial Clusters	Group locations into clusters using algorithms like K-means.	Pollution hotspot zones in a city.

In a property price prediction model, knowing the distance from each property to the nearest school, hospital, or railway station can be a very powerful feature, sometimes more useful than the area of the property itself.

Check Your Progress - 1

a) Fill in the blank: _____ transformations are commonly used to reduce skewness in highly skewed numeric data.

b) Fill in the blank: _____ features are created by combining two or more existing features, like multiplying Price by Quantity to get Total_Sale.

c) Which of the following helps in capturing weekly patterns? (a) Hour of Day (b) Day of Week (c) Year (d) Pin Code

d) What is the primary purpose of creating geospatial features?

e) List any three features you can extract from a single date column.

3.3 Handling Text Data for Feature Engineering

Text data, such as tweets, product reviews, news articles, and chat messages, is everywhere. But machine learning algorithms cannot directly work with text. We must convert text into numbers using a process called text preprocessing followed by vectorisation. This section covers the main techniques in simple language.

3.3.1 Text Cleaning Steps

Before doing anything fancy, we usually clean the text. Common steps include:

- Convert all text to lower case so that Phone, phone, and PHONE become the same word.
- Remove punctuation marks like commas, full stops, and question marks.
- Remove numbers if they are not important.
- Remove stop words. Stop words are very common words like the, is, a, an, of that do not carry much meaning.
- Remove extra whitespace.

3.3.2 Tokenization

Tokenization is the process of breaking text into smaller pieces called tokens. The pieces can be words, sentences, or even smaller parts.

Word-level Tokenization: The most common type. Splits a sentence into individual words. Example: "I love coding" becomes ["I", "love", "coding"].

Sentence-level Tokenization: Splits a paragraph into sentences. Useful in summarisation tasks.

Subword Tokenization: Breaks words into smaller meaningful parts. For example, "playing" can become ["play", "##ing"]. This is used in modern models like BERT.

Hands-on tokenization in Python using NLTK:

```
from nltk.tokenize import word_tokenize, sent_tokenize
text = 'Hello! Welcome to the BCA programme. Coding is fun.'
# Sentence tokenization
print(sent_tokenize(text))
```

```
# Word tokenization
print(word_tokenize(text))
```

3.3.3 Stemming and Lemmatization

Different forms of the same word, like "play", "plays", "playing", "played", all carry similar meaning. To reduce noise, we usually convert them to a single root form.

Stemming: Cuts off endings to find the base form. It is fast but the result may not be a real word. For example, "running", "runner", "ran" may all be reduced to "run". But "studies" may become "studi", which is not a real English word.

Lemmatization: Uses a dictionary to find the proper root word. "better" becomes "good", "running" becomes "run", "studies" becomes "study". It is slower but more accurate.

Example in NLTK:

```
from nltk.stem import PorterStemmer, WordNetLemmatizer
ps = PorterStemmer()
wnl = WordNetLemmatizer()
words = ['running', 'studies', 'better', 'played']
print('Stemming:    ', [ps.stem(w) for w in words])
print('Lemmatized:  ', [wnl.lemmatize(w, pos='v') for w in words])
```

3.3.4 Bag of Words (BoW)

Bag of Words is the simplest way to convert text into numbers. We make a list of all unique words in the dataset, and for each document we count how many times each word appears. The result is a table where each row is a document and each column is a word.

It is called "bag" because the order of words does not matter. The sentence "I love Python" and "Python love I" produce the same vector.

Example: suppose we have these two reviews:

- Review 1: "The phone is great"
- Review 2: "The phone is bad"

The unique words are: the, phone, is, great, bad. The Bag of Words representation is:

Review	the	phone	is	great	bad
Review 1	1	1	1	1	0
Review 2	1	1	1	0	1

In scikit-learn, this is done using CountVectorizer:

```
from sklearn.feature_extraction.text import CountVectorizer
docs = ['The phone is great', 'The phone is bad']
cv = CountVectorizer()
X = cv.fit_transform(docs)
print(cv.get_feature_names_out())
print(X.toarray())
```

3.3.5 TF-IDF

Bag of Words has a problem: very common words like "the" appear in almost every document and get a high count, even though they carry little meaning. TF-IDF (Term Frequency - Inverse Document Frequency) fixes this by giving lower weight to common words and higher weight to rare, meaningful words.

In simple words:

- Term Frequency (TF): how often a word appears in a single document.
- Inverse Document Frequency (IDF): how rare the word is across all documents. A word that appears in every document gets a small IDF; a word that appears in only one document gets a big IDF.
- TF-IDF score = TF multiplied by IDF. Words that appear often in one document but rarely in others get the highest scores.

In scikit-learn, this is done using TfidfVectorizer:

```
from sklearn.feature_extraction.text import TfidfVectorizer
docs = ['The phone is great',
        'The phone is bad',
```

```
    'I love Python']
tfidf = TfidfVectorizer()
X = tfidf.fit_transform(docs)
print(tfidf.get_feature_names_out())
print(X.toarray())
```

TF-IDF is one of the most widely used methods to convert text into numbers and is often used in spam detection, search engines, and document classification.

3.3.6 Word Embeddings

Word embeddings are a more modern way of representing words as numbers. Each word is converted into a list of numbers (a vector) such that similar words get similar vectors. For example, the vectors for "king" and "queen" will be close to each other, and the vector for "car" will be far from both.

Three popular word embedding methods are:

Word2Vec: Trains on large amounts of text and learns vectors so that semantically similar words are close in vector space.

GloVe (Global Vectors): Built from word co-occurrence statistics across a whole text corpus.

FastText: Like Word2Vec but also looks at parts of words (subwords). This makes it good for rare words and spelling variations.

As a BCA student, you do not need to train these models from scratch. Pre-trained word vectors can be downloaded and used directly through libraries like gensim or spaCy.

3.4 Handling Time Series Data

Time series data is data recorded at successive time intervals, like daily stock prices, hourly temperature, monthly sales, or website traffic per minute. Time series feature engineering aims to extract patterns over time so that the model can use them for forecasting.

3.4.1 Time Series Decomposition

A time series can usually be broken down into four parts:

Table 3.5: Components of a Time Series

Component	Description	Example
Trend	The overall direction in which the data is moving over a long time.	Smartphone sales going up year by year.
Seasonality	Patterns that repeat at fixed intervals like daily, weekly, or yearly.	Higher sales of sweets during Diwali every year.
Cyclic	Up-and-down patterns that are not at fixed intervals.	Stock market booms and busts caused by economic cycles.
Residual or Noise	Random variation left after removing the above three.	Sudden drop in app users due to a one-day server outage.

Types of Decomposition

There are two common ways of combining the components:

Additive Model: Y(t) = Trend + Seasonality + Residual. Used when the size of the seasonal effect does not change over time.

Multiplicative Model: Y(t) = Trend x Seasonality x Residual. Used when the seasonal effect grows or shrinks with the trend.

3.4.2 Time-Based Feature Engineering

From a time series, we can create many useful features. The most common ones are listed below.

Table 3.6: Common Time Series Features

Feature	Description	Example
Lag Features	The value at a previous time step (lag-1, lag-2, etc.).	Use yesterday's stock price to predict today's.
Rolling Mean	Average over a moving window (last 7 days, last 30 days).	7-day rolling average of website visits to smooth daily variation.

Feature	Description	Example
Differencing	Subtract the previous value from the current value to remove a trend.	Difference of monthly sales values to make the data stationary.
Day of Week / Weekend	A flag for weekday vs weekend.	Sales spikes on Saturday and Sunday.
Holiday Indicator	A flag for special days (Diwali, Christmas, election day).	Big increase in food orders on holidays.

Hands-On Lag Features in Pandas

```
import pandas as pd
df = pd.DataFrame({
   'Date':  pd.date_range('2024-01-01', periods=7),
   'Sales': [200, 220, 250, 230, 260, 280, 300]
})
df['Sales_Lag_1'] = df['Sales'].shift(1)
df['Rolling_Mean_3'] = df['Sales'].rolling(window=3).mean()
print(df)
```

The shift function moves values down by one row, giving us the previous day's sales as a new feature. The rolling mean smooths out short-term fluctuations and shows the underlying trend.

Check Your Progress - 2

f) Fill in the blank: The technique that gives lower weight to very common words and higher weight to rare meaningful words is called _____.

g) Fill in the blank: Splitting a sentence into individual words is called _____.

h) Which of these is a popular word embedding technique? (a) BoW (b) Word2Vec (c) IQR (d) PCA

i) Name the four components into which a time series is usually decomposed.

j) What is a lag feature in time series? Give one example.

3.5 Feature Selection Techniques

Once we have created many new features, we may end up with too many. Some of them may not be useful, and using too many features can slow down our model and even reduce accuracy. Feature selection is the process of picking only the most useful features and dropping the rest.

The benefits of feature selection are:

- Reduces overfitting by removing noisy features.
- Reduces training time.
- Improves model accuracy by focusing on the right inputs.
- Makes the model easier to understand and explain.

Feature selection methods are usually grouped into three categories: filter, wrapper, and embedded methods.

3.5.1 Filter Methods

Filter methods rank features based on simple statistical scores, without using any machine learning model. They are fast and a good first step in feature selection.

Correlation: Measures how strongly each feature is related to the target. Features with very low correlation are dropped. We also drop features that are very correlated with each other (because they carry the same information).

Chi-square Test: A statistical test used to check the relationship between a categorical feature and a categorical target.

Mutual Information: Measures how much information a feature gives about the target, capturing both linear and non-linear relationships.

Example: predicting house price from features like Area, Number_of_Rooms, Year_Built, Door_Color, and Owner_Name. Correlation analysis will show that Door_Color and Owner_Name have almost no correlation with price, so we drop them.

3.5.2 Wrapper Methods

Wrapper methods try different combinations of features and check which combination gives the best model. They are more accurate than filter methods but much slower because they actually train models multiple times.

Recursive Feature Elimination (RFE): We start with all features, train a model, drop the least important feature, retrain, and repeat. This continues until we reach the desired number of features.

Forward Selection: Start with no features. Add the best one. Add the next best one. Continue until adding more features stops improving accuracy.

Backward Elimination: Start with all features. Drop the worst one. Continue dropping until accuracy starts to fall.

3.5.3 Embedded Methods

Embedded methods perform feature selection as part of the model training itself. They are a smart middle ground between filter and wrapper methods.

Lasso (L1) Regularisation: A type of regression that penalises large coefficients. It often shrinks the coefficients of unimportant features all the way to zero, which effectively removes them.

Ridge (L2) Regularisation: Penalises large coefficients but does not usually drive them to zero. It reduces the impact of less important features rather than removing them.

Tree-based Importance: Tree algorithms like Random Forest and Gradient Boosting can rank features by importance. We keep only the top N features.

Table 3.7: Comparing Feature Selection Methods

Method Type	Speed	Accuracy	Example
Filter	Very fast	Moderate	Correlation, Chi-square
Wrapper	Very slow	High	RFE, Forward Selection
Embedded	Medium	High	Lasso, Ridge, Random Forest importance

3.6 Feature Scaling and Normalization

We already discussed scaling in Unit 2. Here we focus on its specific role in feature engineering and on which algorithms need it.

3.6.1 Why Scaling Matters

If two features have very different scales, like Age (18 to 70) and Salary (10,000 to 10,00,000), the larger one will dominate algorithms that use distance, like K-Nearest Neighbors and clustering. Scaling brings all features to a comparable range and ensures that each one contributes fairly.

3.6.2 Common Scaling Methods (Quick Recap)

- Min-Max Scaling: scales values to the range 0 to 1.
- Standardization (Z-score): centres data so that mean is 0 and standard deviation is 1.
- Robust Scaling: uses median and IQR; suitable when there are many outliers.

3.6.3 Effect on Different Algorithms

Not every algorithm needs scaling. The table below summarises which algorithms need it and which do not.

Table 3.8: Scaling Requirements by Algorithm

Algorithm Group	Needs Scaling?	Reason
K-Nearest Neighbors, K-Means, Clustering	Yes	They calculate distances; large-scale features will dominate.
Linear Regression, Logistic Regression, SVM	Yes	Sensitive to feature magnitude; scaling helps gradient descent converge faster.
Neural Networks (Deep Learning)	Yes	Faster training, more stable gradients.
Decision Trees, Random Forest, Gradient Boosting	No	Tree splits are based on values, not distances; scaling does not affect splits.

3.6.4 Use Cases for Different Scaling Techniques

Standardization in Regression: When we predict, say, weight from height, standardisation makes the regression coefficients easier to interpret.

Min-Max Scaling in Image Processing: Image pixel values are usually scaled to the range 0 to 1 before being fed to a neural network.

Robust Scaling for Financial Data: Where there are many extreme values, robust scaling protects against the effect of outliers.

Check Your Progress - 3

k) True or False: Filter methods evaluate features without using any machine learning model.

l) True or False: Tree-based algorithms like Random Forest are highly sensitive to feature scaling.

m) What is the main purpose of feature selection in data science?

n) When would you choose Standardization over Min-Max scaling?

o) Name any two embedded methods of feature selection.

3.7 Let us sum up

In this unit, we covered the essentials of feature engineering, the most creative and impactful step in any data science project. We learned how to create new features using mathematical transformations and interaction features, and how to extract useful features from date, time, and location data. We discussed text data and saw how techniques like tokenization, stemming, lemmatization, Bag of Words, TF-IDF, and word embeddings convert raw text into numbers. We then looked at time series data and learned about decomposition (trend, seasonality, cyclic, residual) and time-based features like lag and rolling mean. Finally, we covered feature selection methods (filter, wrapper, and embedded) and feature scaling for different machine learning algorithms. With these skills, you can take any raw dataset and build a much richer, smarter set of features that will help your models perform better.

3.8 Check your progress: Possible Answers

a) Logarithmic
b) Interaction
c) (b) Day of Week
d) Geospatial features capture location-based patterns. They help models understand how the location of an event, customer, or property influences the target variable, such as the price of a house or the demand for a service in a particular area.
e) Three features from a date column: Year, Month, Day_of_Week (other valid answers: Hour, Is_Weekend, Quarter).
f) TF-IDF
g) Tokenization
h) (b) Word2Vec
i) Trend, Seasonality, Cyclic, and Residual (Noise)
j) A lag feature is the value of a variable at a previous time step. Example: using yesterday's stock price as a feature to predict today's price.
k) True
l) False. Tree-based algorithms are usually NOT sensitive to scaling, because they split based on values, not distances.
m) The main purpose of feature selection is to keep only the most useful features and drop the rest, which reduces overfitting, speeds up training, and improves model accuracy and interpretability.
n) We choose Standardization when the algorithm assumes data centred around 0 with unit variance, such as linear regression, logistic regression, and SVM, especially when the data is roughly normally distributed.
o) Two embedded methods: Lasso (L1) and Ridge (L2) regularisation. Tree-based feature importance from Random Forest is also a valid answer.

3.9 Further Readings

1. Feature Engineering for Machine Learning, Alice Zheng and Amanda Casari, O'Reilly Media.
2. Hands-On Data Preprocessing in Python, Roy Jafari, Packt Publishing.

3. Online tutorials on Pandas time series at the official Pandas website.
4. scikit-learn user guide on feature selection and preprocessing

3.10 Assignments

1. Define feature engineering. Explain why it is one of the most important steps in machine learning, with at least three reasons.
2. List and explain any three mathematical transformations used in feature creation, with one example for each.
3. Explain the role of interaction features in feature engineering. Provide three examples from real-life data.
4. Suppose you have a single column called Order_DateTime. List at least five new features you can create from it, and briefly explain the use of each.
5. Discuss the process of tokenization, stemming, and lemmatization with one example each.
6. Compare Bag of Words and TF-IDF as text vectorisation methods. Which one would you prefer for a spam classification project? Justify.
7. Define time series decomposition and explain each of its four components with one real-life example.
8. What are lag features and rolling mean features in time series? Why are they useful for forecasting?
9. Describe filter, wrapper, and embedded methods of feature selection. Give one example of each and explain its main strength.
10. Practical Assignment: Take any small dataset of your choice (it can be a CSV from Kaggle or a dataset you create). Apply at least four different feature engineering techniques (one mathematical transformation, one interaction feature, one date-time feature, and one feature selection step) using Pandas. Submit the code and explain the new features in 5-6 lines.

Unit-4: Data Integration and Merging

Unit Structure

4.0 Learning Objectives

After studying this unit student should be able to:

- Explain what data integration means and why it is important in data science projects.
- Identify various types of data integration techniques such as schema integration, record linkage, and deduplication.
- Describe the ETL (Extract, Transform, Load) process and the role of data warehousing.
- Apply common data merging techniques such as join operations and concatenation, and understand the difference between them.
- Combine structured data from databases and CSV files using SQL and Pandas.
- Understand how unstructured data like text, images, and videos can be integrated.
- Perform data validation and consistency checks after merging to ensure data quality.

4.1 Introduction to Data Integration

Real-world data is rarely stored in a single place. Imagine a college that has student information in one database, fee details in a second database, and attendance records in a third system, and library issue records in a fourth. To get a complete picture of any one student, we must combine information from all these places. This act of combining data from multiple sources into a single unified view is called data integration.

Data integration is the foundation of any large-scale analytics project. Without it, we are like a doctor who can see only one part of the patient's medical history at a time. With it, we get the complete picture and can make far better decisions.

Why Data Integration Matters

- It improves decision-making by combining information that was earlier scattered.
- It gives a more complete view of the topic, customer, or business.
- It helps in finding inconsistencies because the same field may be stored differently in different systems.
- It supports large-scale projects, where data must come from many sources.
- It enables advanced analytics and machine learning that need rich, combined datasets.

A Simple Example for BCA Students

Suppose you are building a college dashboard. You want to show, for every student, their personal details, marks, attendance, fee status, and library activity. The data for this comes from four different systems. Data integration is the work of bringing all these together so that the dashboard shows a single, complete profile.

Common Challenges in Data Integration

Combining data sounds simple, but in practice it is one of the most challenging steps in any data project. The main difficulties are:

Structural Heterogeneity: Different sources may use different formats. One system may store dates as DD-MM-YYYY, another as YYYY-MM-DD. One database may have a column called Roll_No, another may call it Enrolment_ID.

Semantic Heterogeneity: The same word can mean different things in different systems. For example, the column Status in one database may mean active or inactive, while in another it may mean pending or cleared.

Data Quality Issues: Different sources may have different levels of accuracy, completeness and consistency.

Scale: The amount of data may be huge, requiring powerful infrastructure.

Privacy and Security: Some data is sensitive (medical, financial). Integrating it across systems must follow privacy laws like the Indian DPDP Act.

4.2 Types of Data Integration

Different projects use different integration approaches, depending on size, urgency, and infrastructure. The main types are described below.

4.2.1 Data Consolidation

In data consolidation, data from multiple sources is physically moved into a single central repository, like a data warehouse. This central repository becomes the single source of truth. Most organisations use this approach for analytics.

4.2.2 Data Federation (Virtual Integration)

In data federation, data is not physically moved. Instead, a virtual layer is created on top of the various sources. When a user runs a query, the system internally fetches data from each source

on the fly and presents a unified result. This is useful when data cannot be moved due to legal or technical reasons.

4.2.3 Data Propagation

In data propagation, changes made in one system are quickly copied to others, often in real time. This is used in apps that need to be in sync, like a banking system where a deposit in one branch must be visible immediately at all branches.

4.2.4 Data Virtualisation

Similar to data federation, data virtualisation gives applications a unified view of data across systems without copying. It is becoming popular for cloud-based applications.

Table 4.1: Comparing Types of Data Integration

Type	Data Movement	When to Use
Consolidation	Physical copy into a central place	Standard analytics and reporting
Federation	No movement, virtual view	When sources cannot be moved (legal, size)
Propagation	Real-time copying	Systems that must stay in sync
Virtualisation	On-demand virtual view	Cloud applications, mixed environments

4.2.5 Schema Integration

Schema integration is the process of bringing different data schemas (structures) into a single unified schema. It is essential when data from multiple databases is being combined.

It involves three main steps:

Schema Matching: Map columns from different sources that mean the same thing. For example, Roll_No in one source matches Enrolment_ID in another.

Schema Transformation: Apply rules to convert data into a common format. Rename columns, change data types, and standardise units.

Schema Merging: Combine the matched schemas into one structure that everyone uses going forward.

4.2.6 Record Linkage and Deduplication

When the same person, product, or event is recorded in two different systems, integration creates duplicates. Record linkage finds such duplicates and links them, while deduplication merges them into a single record.

For example, the same customer may exist in the sales database as Asha Patel and in the marketing database as A. Patel. Record linkage figures out that they are the same person, and deduplication merges the two records.

Record linkage is very important in fields like:

- Customer analytics: combining records of the same customer across departments.
- Health care: linking patient records across hospitals.
- Fraud detection: spotting the same person registered under multiple identities.

4.2.7 Data Warehousing and ETL

A data warehouse is a large central repository where data from multiple sources is brought together for analysis and reporting. The process used to fill a data warehouse is called ETL (Extract, Transform, and Load).

Extract: Pull data from each source system, such as databases, CSV files, APIs, and log files.

Transform: Clean and convert the data to a common format. This is where preprocessing, schema integration, and validation happen.

Load: Save the cleaned, integrated data into the data warehouse, ready for analysis.

Many organisations use ETL pipelines that run automatically every night, so that the next morning, analysts get fresh, integrated data ready for use. Modern variants like ELT (Extract, Load, Transform) are also popular, especially for big data systems.

Check Your Progress - 1

a) Fill in the blank: _____ integration physically moves data from multiple sources into a central repository, creating a single source of truth.

b) Fill in the blank: In the ETL process, the _____ phase is where data is cleaned and converted into a suitable common format.

c) Which type of data integration uses a virtual layer instead of physically moving data?

d) What is the purpose of record linkage in data integration?

e) Define a data warehouse in your own words.

4.3 Techniques for Data Merging

Data merging is the practical act of combining two or more datasets into one. The main techniques used are join operations (combining tables based on a common key) and concatenation (stacking tables together).

4.3.1 Join Operations

Join operations combine rows from two or more tables based on a related column, such as a customer ID or order number. The type of join we choose decides which rows are kept from each table.

Suppose we have two simple tables:

Customers table:

CustomerID	Name	City
C1	Asha	Anand
C2	Bhavin	Surat
C3	Chirag	Rajkot
C4	Disha	Vadodara

Orders table:

OrderID	CustomerID	Amount
O1	C1	1500
O2	C2	800
O3	C2	2000
O4	C5	500

Notice that customer C5 in the Orders table does not exist in the Customers table, and customers C3 and C4 have not placed any orders.

Inner Join

An inner join returns only the rows where there is a match in both tables. In our example, an inner join on CustomerID gives only the rows for C1 (with order O1) and C2 (with orders O2 and O3). Customers C3 and C4 are dropped because they have no orders, and order O4 is dropped because customer C5 does not exist in the Customers table.

Left Join (Left Outer Join)

A left join returns all rows from the left table, plus matching rows from the right table. Where there is no match, the right table columns are filled with NaN. So a left join of Customers and Orders gives all four customers; C3 and C4 will have NaN in the order columns.

Right Join (Right Outer Join)

A right join is just the mirror of a left join. It returns all rows from the right table, plus matching rows from the left table. Order O4 (for non-existent customer C5) is kept, with NaN in the customer columns.

Full Outer Join

A full outer join keeps everything from both tables. Rows that don't match get NaN in the missing columns. This is useful when we don't want to lose anything.

Table 4.2: Quick Comparison of Joins

Join Type	What It Keeps
Inner Join	Only rows that match in both tables
Left Join	All rows from the left + matched rows from the right
Right Join	All rows from the right + matched rows from the left
Full Outer Join	All rows from both tables, with NaN where no match

Hands-On with Pandas

Pandas provides the merge function for joins. Let us see all four join types in action.

```
import pandas as pd
customers = pd.DataFrame({
```

```
    'CustomerID': ['C1', 'C2', 'C3', 'C4'],
    'Name': ['Asha', 'Bhavin', 'Chirag', 'Disha'],
    'City': ['Anand', 'Surat', 'Rajkot', 'Vadodara']
})
orders = pd.DataFrame({
    'OrderID': ['O1', 'O2', 'O3', 'O4'],
    'CustomerID': ['C1', 'C2', 'C2', 'C5'],
    'Amount': [1500, 800, 2000, 500]
})
# Inner Join
inner = pd.merge(customers, orders, on='CustomerID', how='inner')
# Left Join
left = pd.merge(customers, orders, on='CustomerID', how='left')
# Right Join
right = pd.merge(customers, orders, on='CustomerID', how='right')
# Full Outer Join
outer = pd.merge(customers, orders, on='CustomerID', how='outer')
print(inner)
print(left)
print(right)
print(outer)
```

The on parameter tells Pandas which column to join on, and how decides the type of join. By trying out the four versions, you can clearly see which rows are kept and dropped each time.

Joins in SQL

In databases, joins are done using SQL. The same logic applies.

```
-- Inner join
SELECT c.CustomerID, c.Name, o.OrderID, o.Amount
FROM Customers c
```

```
INNER JOIN Orders o
ON c.CustomerID = o.CustomerID;


-- Left join
SELECT c.CustomerID, c.Name, o.OrderID, o.Amount
FROM Customers c
LEFT JOIN Orders o
ON c.CustomerID = o.CustomerID;
```

As a BCA student, you should be comfortable with both Pandas joins and SQL joins, because both are very commonly used in industry.

4.4.2 Concatenation

Concatenation is used when datasets have a similar structure and we want to stack them together, either vertically (more rows) or horizontally (more columns).

Vertical Concatenation

Vertical concatenation stacks tables on top of each other, adding more rows. It is used when datasets share the same columns but contain different records.

For example, suppose you have monthly sales data in separate files: jan_sales.csv, feb_sales.csv, mar_sales.csv. All three have the same columns. Vertically concatenating them gives a single first-quarter sales dataset.

```
import pandas as pd
jan = pd.read_csv('jan_sales.csv')
feb = pd.read_csv('feb_sales.csv')
mar = pd.read_csv('mar_sales.csv')
q1_sales = pd.concat([jan, feb, mar], axis=0)
print(q1_sales.shape)
```

The axis=0 means stack vertically (add rows). It is the default, but it's good practice to write it explicitly.

Horizontal Concatenation

Horizontal concatenation joins tables side by side, adding more columns. It is useful when both tables have the same primary key but contain different attributes.

For example, you may have one dataset with student demographic details and another with their academic performance, both indexed by Roll_No. Horizontal concatenation combines them into a single rich dataset.

```
# Horizontal concatenation
combined = pd.concat([demographics, academics], axis=1)
```

Note: For most real cases of combining different attributes by key, a merge (join) is safer than horizontal concatenation, because merge handles the case where the order of rows is different.

Union and Append

Union: In SQL, UNION combines the rows of two SELECT queries, removing duplicates by default. UNION ALL keeps duplicates.

Append: In Pandas, append used to add rows from one DataFrame to another. It has been deprecated and replaced by pd.concat.

```
-- SQL example of UNION
SELECT name FROM Customers_2023
UNION
SELECT name FROM Customers_2024;
```

4.4 Data Integration from Structured and Unstructured Sources

Data comes in many shapes. We can broadly group it into structured and unstructured.

Structured Data: Has a fixed format with rows and columns. Stored in databases, spreadsheets, or CSV files.

Unstructured Data: Does not have a fixed format. Includes text, images, audio, video, and PDFs.

4.4. 1 Integrating Structured Data

Structured data is the easiest to integrate because it has clear columns and types. Common sources are databases and CSV files.

From Databases

Database integration uses SQL operations like JOIN, UNION and INSERT INTO. Multiple databases can also be linked through ETL tools.

From CSV Files

CSV files are very popular because they are simple text files supported by every tool. Pandas can read and merge multiple CSV files easily.

```
import pandas as pd
students = pd.read_csv('students.csv')
marks = pd.read_csv('marks.csv')
merged = pd.merge(students, marks, on='Roll_No', how='left')
merged.to_csv('students_with_marks.csv', index=False)
```

In just five lines, we have read two CSV files, joined them on Roll_No, and saved the merged result as a new CSV file. This is a very common task in real projects.

ETL for Structured Data

In larger projects, the entire data pipeline is automated. Data is extracted from many databases and CSV files (Extract), cleaned and reformatted (Transform), and loaded into a single warehouse (Load). Tools like Apache Airflow, Talend, and Informatica are used to schedule and run these pipelines automatically.

4.5.2 Integrating Unstructured Data

Unstructured data is harder to handle because there are no rows and columns. We need special techniques to extract useful information first, and then store that information in a structured form for integration.

Text Data

Text data, such as customer reviews, support emails, and tweets, can be processed using NLP (Natural Language Processing) techniques. We can:

- Apply preprocessing (tokenization, stop word removal, stemming) as discussed in Unit 3.
- Convert text to numbers using Bag of Words, TF-IDF or word embeddings.

- Extract entities like names, places, organisations, and dates.
- Perform sentiment analysis to find whether the text is positive, negative or neutral.

Image Data

Image integration usually involves:

- Image recognition: identifying what is in the image (cat, car, person, building).
- Image tagging: adding labels to images for easier search.
- Feature extraction using deep learning models like Convolutional Neural Networks (CNNs).
- Storing image metadata such as size, location, time, and detected objects in a structured database.

Video and Audio Data

Video data is essentially a sequence of images plus an audio track. Common integration tasks include:

- Speech-to-text conversion using tools like Google Speech API or open-source tools like Whisper.
- Object detection and tracking across frames.
- Sentiment from voice tone for call centre quality analysis.

Once we extract structured information from text, images, audio or video, we can integrate it with other structured data using normal joins and concatenation. This is how modern systems like search engines, recommendation engines, and digital assistants work.

Check Your Progress - 2

f) True or False: An inner join keeps all rows from the left table even when there is no match in the right table.

g) Which Pandas function is used to perform a join between two DataFrames?

h) Differentiate between vertical concatenation and horizontal concatenation in two lines.

i) Name two NLP techniques used to integrate text data.

j) Give one real-life example each of structured and unstructured data.

4.5 ETL (Extract, Transform, Load) Frameworks

ETL frameworks are tools that help us build, schedule, and monitor data pipelines. They make it easy to move data from one place to another while cleaning and transforming it on the way.

4.5.1 Common ETL Tools

Some popular ETL tools used in industry are:

Table 4.3: Popular ETL Tools

Tool	Type	What It Does
Apache Airflow	Open Source	Schedules and runs data pipelines using Python; widely used in industry.
Apache NiFi	Open Source	Visual tool for building real-time data pipelines, especially for streaming and IoT.
Talend	Open Source / Commercial	Drag-and-drop ETL with hundreds of pre-built connectors.
Informatica PowerCenter	Commercial	Industry-grade ETL platform used by large enterprises.
AWS Glue	Commercial (Cloud)	Fully managed ETL service in Amazon Web Services.
Microsoft SSIS	Commercial	ETL tool that comes with SQL Server.

4.5.2 Workflow Automation

ETL pipelines often run on a schedule, for example every night at 2 am. Workflow automation tools take care of:

- Running each step in the right order.
- Retrying steps that fail because of temporary problems.
- Sending alerts to data engineers when something goes wrong.
- Logging everything so that issues can be debugged later.
- Allowing easy expansion when new sources are added.

With proper workflow automation, a team of three or four engineers can manage data pipelines that handle terabytes of data every day.

4.6 Data Validation and Integrity Post-Merging

After merging data from multiple sources, we must verify that the result is correct, consistent, and trustworthy. This step is called data validation, and it is one of the most important parts of any integration project.

4.6.1 Data Consistency Checks

Consistency checks make sure data follows expected rules. The most common types are listed below.

Table 4.4: Types of Consistency Checks

Check	Description	Example
Range Check	Values must lie within a known acceptable range.	Age between 0 and 120, marks between 0 and 100.
Uniqueness Check	Certain fields must contain unique values.	Roll_No, Aadhaar Number, OrderID.
Format Check	Values must follow a specific format.	Email contains @ sign; phone has 10 digits.
Referential Integrity	Foreign key in one table must exist in the other.	Every CustomerID in Orders must exist in Customers.
Completeness Check	Required fields are not missing.	Every student must have a Name and a Roll_No.

Manual vs Automated Checks

- Automated checks: Used for large datasets. Scripts or ETL tools quickly find inconsistencies.
- Manual checks: Used for sensitive small datasets where human judgement is needed.

4.6.2 Auditing Methods

Auditing is a thorough way of comparing the merged data with the original data to make sure nothing has been lost or distorted during integration. Common auditing methods are:

Source-to-Target Validation: Compare each source table with the merged result to confirm no data was lost or changed accidentally.

Field-by-Field Validation: Check each column to make sure values match expected criteria.

Statistical Auditing: Compare summary statistics (sum, mean, count) before and after merging. They should match closely.

Record Count: Count the number of records before and after to detect missing or duplicate rows.

4.6.3 Data Validation Techniques

Beyond auditing, we use specific techniques to validate the merged data. Some of the popular ones are:

Cross-Validation: Cross-check data with another reliable source. Useful for financial and transactional data where two systems should always match.

Checksum Validation: Use hash functions to check that data was not accidentally changed during transfer.

Data Profiling: Analyse the structure and content of data to find anomalies. Tools like Pandas Profiling or ydata-profiling can generate detailed reports automatically.

Data Quality Rules: Define rules for each field, such as required fields, value ranges, and formats. ETL tools can apply these rules automatically.

4.6.4 Hands-On Validation in Pandas

Below is a small Python program that demonstrates several validation checks on a merged DataFrame.

```
import pandas as pd
# Suppose this is the merged DataFrame
df = pd.DataFrame({
   'StudentID': ['S1', 'S2', 'S3', 'S4', 'S2'],
```

```
    'Name': ['Asha', 'Bhavin', 'Chirag', 'Disha', 'Bhavin'],
    'Marks': [78, 85, 200, 90, 85]
})
# 1. Range check on Marks
invalid = df[(df['Marks'] < 0) | (df['Marks'] > 100)]
print('Out of range marks:')
print(invalid)
# 2. Uniqueness check on StudentID
duplicates = df[df.duplicated(subset='StudentID', keep=False)]
print('Duplicate StudentIDs:')
print(duplicates)
# 3. Completeness check
missing = df.isna().sum()
print('Missing values per column:')
print(missing)
```

This simple example covers three of the most common validation checks. In a real ETL pipeline, hundreds of such checks may run automatically every night.

4.6.5 Continuous Monitoring

Validation is not a one-time activity. Data sources change, new errors creep in, and the volume of data grows. Continuous monitoring keeps track of data quality over time.

- Schedule regular audits, for example every week or every month.
- Set up automated checks that alert the team when a rule is violated.
- Maintain dashboards that show key data quality metrics, like percentage of missing values, number of duplicate rows, and number of out-of-range values.

Check Your Progress - 3

k) True or False: Apache NiFi is mainly used for creating and managing real-time data streaming pipelines.

l) Fill in the blank: In ETL, the step where data is converted into the target format is called the _____ step.

m) What is the main purpose of workflow automation in ETL frameworks?

n) True or False: Range checks ensure that all numerical values fall within predefined acceptable limits.

o) Fill in the blank: _____ checks ensure that identifiers like User_ID and Order_ID are not duplicated in the merged dataset.

4.7 A Mini End-to-End Example

Let us put together everything we have learned in this unit through a small example. Suppose we have:

- students.csv with columns Roll_No, Name, Branch, Year.
- marks.csv with columns Roll_No, Subject, Marks.
- attendance.csv with columns Roll_No, Subject, Total_Classes, Attended_Classes.

We want to build one combined dataset that has, for every student, their personal details, average marks, and average attendance percentage. Then we want to validate the result.

```
import pandas as pd
# 1. EXTRACT: read the source files
students   = pd.read_csv('students.csv')
marks      = pd.read_csv('marks.csv')
attendance = pd.read_csv('attendance.csv')
# 2. TRANSFORM: aggregate and merge
avg_marks = marks.groupby('Roll_No')['Marks'].mean().reset_index()
avg_marks.rename(columns={'Marks': 'Avg_Marks'}, inplace=True)
attendance['Att_Pct'] = (
   attendance['Attended_Classes'] / attendance['Total_Classes'] * 100
)
avg_att = attendance.groupby('Roll_No')['Att_Pct'].mean().reset_index()
```

```
avg_att.rename(columns={'Att_Pct': 'Avg_Attendance'}, inplace=True)
# Merge step by step
temp = pd.merge(students, avg_marks, on='Roll_No', how='left')
final = pd.merge(temp, avg_att, on='Roll_No', how='left')
# 3. VALIDATE: simple checks
print('Rows in students :', len(students))
print('Rows in final    :', len(final))
print('Missing values   :')
print(final.isna().sum())
# 4. LOAD: save the merged file
final.to_csv('student_summary.csv', index=False)
```

Step by step, this small program covers all four parts of ETL. We extracted three CSV files, transformed them by computing averages and renaming columns, merged them with two left joins, validated row counts and missing values, and finally saved the integrated dataset. With this template, you can handle a wide range of real integration tasks

4.8 Let Us Sum Up

In this unit, we explored data integration and merging, two skills you will use in almost every real-world data project. We started with the meaning and importance of data integration, and looked at the main challenges like structural and semantic heterogeneity. We discussed the main types of integration, including consolidation, federation, propagation, and virtualisation, and the special role of schema integration, record linkage, and ETL. We then learned the practical techniques of data merging through join operations (inner, left, right, full outer) and concatenation (vertical and horizontal), with hands-on examples in Pandas and SQL. We extended this to integrating structured data from databases and CSV files, as well as unstructured data like text, images and videos. We covered popular ETL tools and the role of workflow automation. Finally, we looked at how to validate merged data using consistency checks, auditing, data profiling, and continuous monitoring. With these skills, you can confidently combine data from many sources into one clean, reliable dataset ready for analysis.

4.9 Check your progress: Possible Answers

a) Consolidation
b) Transform
c) Data Federation (also called virtual integration)
d) Record linkage identifies and matches records from different sources that refer to the same real-world entity (such as the same customer or product), so that we do not end up with duplicate records in the integrated dataset.
e) A data warehouse is a large central repository where cleaned and integrated data from multiple sources is stored together for the purpose of analysis, reporting, and decision-making.
f) False. An inner join keeps only the rows that have matching keys in both tables. The left join keeps all rows from the left table.
g) pd.merge
h) Vertical concatenation stacks tables with the same columns on top of each other, adding more rows. Horizontal concatenation joins tables side by side, adding more columns.
i) Two NLP techniques: tokenization and TF-IDF (other valid answers: stemming, lemmatization, sentiment analysis, named entity recognition).
j) Structured data: the marks table in a college database. Unstructured data: a customer review on an e-commerce site, or a video uploaded to YouTube.
k) True
l) Transform
m) Workflow automation in ETL frameworks is used to schedule, run, and monitor data pipelines automatically, retry failed steps, send alerts when something goes wrong, and ensure that data is delivered on time with minimal human intervention.
n) True
o) Uniqueness

4.10 Further Readings

1. Hands-On Data Preprocessing in Python, Roy Jafari, Packt Publishing.
2. Data Integration: A Theoretical Perspective, Maurizio Lenzerini (research paper, freely available online).

3. Online tutorials on Pandas merge and concat at the Pandas official documentation.
4. Online tutorials on SQL joins at W3Schools, GeeksforGeeks, and the SQLZoo website.

4.11 Assignments

1. Define data integration. Discuss its importance with at least three real-life examples from a college or business environment.
2. Explain any three challenges of data integration with one example each.
3. Differentiate between data consolidation, data federation, data propagation, and data virtualisation in your own words.
4. Describe the ETL process and explain each of its three steps with a small example.
5. Explain inner join, left join, right join, and full outer join with a clear small example using two tables (Customers and Orders).
6. Differentiate between vertical concatenation and horizontal concatenation. Give one real-life situation where each would be useful.
7. Compare and contrast the integration of structured data (databases, CSV files) and unstructured data (text, images, videos). Discuss the unique challenges of each and give possible solutions.
8. Describe at least four types of consistency checks performed after merging data, with one example each.
9. Practical Assignment: Create three small CSV files: students.csv, marks.csv, and attendance.csv (with at least 10 rows each). Write a Python program using Pandas that (a) performs a left join of all three on Roll_No, (b) calculates Average Marks and Average Attendance per student, (c) checks for any missing values, and (d) saves the merged file as student_summary.csv. Submit the code and the output.
10. Choose any one ETL tool from Apache Airflow, Talend, AWS Glue, or SSIS. Write a one-page write-up about its main features, advantages, and a real-life use case where it is used.

Block-3

Statistical Foundations and Analytics

Unit-1: Basic Mathematic

Unit Structure

1.0 Learning Objectives

After studying this unit student should be able to:

- Understand why mathematics is important in Data Science
- Understand number systems and basic operations in context to Data Science
- Understand basic algebra concepts for Data Science
- Understand functions and graphs for Data Science

1.1 Introduction

Data science is built on mathematics, which is essential to comprehending, evaluating, and interpreting data. Organizations extensively rely on mathematical ideas in today's data-driven environment to make well-informed decisions, spot patterns, and forecast future trends. Fundamentally, mathematics offers a methodical approach to problem-solving and thought. It facilitates the use of relationships, symbols, and numbers to depict real-world scenarios. Raw data is frequently complicated and disorganized in data research. We can extract valuable insights from this data by cleaning, transforming, and analyzing it using mathematical methods. Applying algorithms and correctly interpreting findings become challenging without a fundamental understanding of mathematics. Managing numerical data is one of mathematics' primary functions in data science. Data analysis often uses concepts like averages, percentages, ratios, and arithmetic operations. For instance, basic mathematical operations are needed to compare values across datasets, calculate the average performance of pupils, and determine growth rates in business. These fundamental skills serve as the building blocks for more advanced techniques. Another crucial element of fundamental mathematics that aids in establishing correlations between variables is algebra. Numerous issues in data science need the estimation or prediction of unknown values. We can represent and systematically tackle these problems using algebraic expressions and equations. For example, regression analysis and predictive modelling both make extensive use of linear equations. Graphs and functions are important tools for visualizing and comprehending the relationships between variables. Graphs offer a visual depiction of this relationship, whereas functions explain how one number changes in relation to another. In data science, trends, patterns, and anomalies in data can be found using graphical representations including scatter plots, bar graphs, and line charts. This facilitates the interpretation and communication of complex information.

The foundation of data science is fundamental mathematics. It offers the methods and resources needed to evaluate data, create models, and extract significant insights. In addition to improving analytical thinking, a solid mathematical foundation helps students comprehend increasingly complex data science concepts. Therefore, any student hoping to pursue a career in data science must grasp these fundamental ideas.

1.2 Number Systems and Basic Operations for Data Science

Working efficiently with data begins with an understanding of number systems and fundamental operations. Nearly all tasks in data science, including data cleaning, analysis, and model construction, rely on numerical values. Students who have a solid foundation in number systems are better able to compute and analyze data.

Based on their characteristics, numbers are categorized into many groups in mathematics. Because different forms of data employ different sorts of numbers, these classifications are crucial.

(a) Natural Numbers : Counting numbers beginning with 1 are known as natural numbers. For instance, 1, 2, 3, 4, 5,

Utilization in Data Science: used to count discrete values like: The quantity of pupils in a class, quantity of goods sold etc.

(b) Whole Numbers : Natural numbers and zero are examples of whole numbers. For instance, 0, 1, 2, 3, 4,

Applications: Number of errors in a program, Number of absent students etc.

(c) Integers: Positive, negative, and zero numbers are all considered integers.

For instance: -3, -2, -1, 0, 1, 2, 3

Applications: Profit or Loss Calculation, Temperature Reading

(d) Rational Numbers : numbers that can be expressed as fractions (p/q).

For instance, 1/2, 3/4, and 0.75

Applications: Ratios,Probabilities, percentages etc.

(e) Irrational Numbers: Numbers that cannot be expressed as fractions and have non-repeating decimals.

Examples: $\sqrt{5}$, π

Applications:

Used in advanced calculations such as:

Geometry

Machine learning algorithms (distance calculations)

(f) Real Numbers

All rational and irrational numbers together form real numbers.

Examples: -5, 0, 2.5, √3

Applications:

Most datasets in data science contain real numbers, such as:

- Height, weight, temperature
- Financial data

Numbers are more than simply abstract mathematical concepts in data science; computers store, process, and manipulate them in certain ways. Because it directly impacts the precision, effectiveness, and dependability of data analysis, it is crucial to comprehend how numbers are represented in computing. Computers employ binary (base-2) systems for all operations, whereas humans naturally use decimal (base-10). The binary system, which consists of just two digits—0 and 1—is the most basic way that computers express numbers. We refer to each digit as a bit (binary digit). Larger units like bytes, which can represent a variety of values, are created by combining these bits. For instance, the binary representation of the decimal number 5 is 101. All forms of data, whether textual, visual, or numerical, are eventually transformed into binary form for processing and storage in data science. Humans use the decimal system to communicate with data, even if computers use binary as their core language. As a result, decimal and binary representations are constantly being converted. Programming languages and software tools handle this conversion automatically, but knowing it aids in error troubleshooting and computation optimization. For example, how numbers are encoded in memory affects how well they are processed and stored when working with enormous datasets. The representation of real numbers (numbers with decimal points) using floating-point representation is another crucial idea. A sign, a mantissa (or significand), and an exponent make up the format used to store floating-point numbers, which is comparable to scientific notation. For instance, a floating-point representation of the value 3.14 is possible. This technique makes it possible for computers to efficiently handle both very big and very small numbers, which is particularly helpful for machine learning algorithms and scientific computations. However, accuracy error is a limitation introduced by floating-point encoding. Certain decimal values cannot be precisely represented in binary form because computers store numbers in a finite number of bits. For instance, when numbers like 0.1 or 0.2 are kept in memory, they may have little errors. These tiny mistakes can add up and have an impact on

data science outcomes, particularly in large-scale or repeated computations. Fixed-size binary formats are used to express integers in addition to floating-point numbers. Typically, computers employ integer formats like 8-bit, 16-bit, 32-bit, or 64-bit. The range of values that can be stored is determined by these formats. A 32-bit integer, for instance, can hold values in the approximate range of −2 billion to +2 billion. In data science, selecting the right data type is crucial to striking a balance between memory utilization and computing efficiency. Signed and unsigned numbers are another idea in number representation. Unsigned numbers only show non-negative values, whereas signed numbers include both positive and negative values. To effectively store negative numbers, computers employ strategies like two's complement representation. In applications like financial analysis, where both profits and losses must be represented, this is especially helpful. In computer languages like Python, R, or SQL, data scientists also come across several numerical data kinds. Integers, floating-point numbers, and occasionally higher-precision formats like decimal are among them. Every kind has a different precision level and storing method. When working with enormous datasets, choosing the right data type is crucial to ensuring accurate computations and optimal performance.

Check Your Progress-1

1. Why mathematics is important in Data Science?
2. What do you mean by natural number? Give at least two examples of it.
3. Give at least two applications of rational numbers in Data Science.
4. Computers employ _____ systems for all operations.
5. Which numerical data are considered in programming languages of Data
6. Science?

1.3 Algebra for Data Science

One of the most crucial areas of mathematics for data science is algebra since it offers a methodical technique to depict the relationships between variables. To put it simply, algebra enables us to convert real-world issues into mathematical equations and expressions. Understanding algebra is crucial since many data science methods, such as machine learning models, rely on algebraic ideas.

Variables and constants are fundamental to algebra. A constant is a stable value, whereas a variable is a symbol (often represented by letters like x, y, and z) that denotes an uncertain or changeable value. Variables are frequently used in data science to represent characteristics or

properties in a dataset. For instance, variables in a student performance dataset could include study hours, attendance, and grades. In equations, constants can stand in for fixed quantities like thresholds or coefficients. Algebra is an effective tool for simulating real-world scenarios because of its capacity to generalize values using variables.

Variables, constants, and mathematical operations like addition, subtraction, multiplication, and division are all combined in algebraic expressions. For instance, a connection where the value depends on x is represented by an expression such as $2x + 5$. Relationships between input variables and outputs are described by such statements in data science. In business analytics, for example, a basic cost function may be expressed as $Cost = 50x + 200$, where x represents the quantity of units produced. Analyzing data requires an understanding of how to create and work with such expressions.

Linear equations, which involve variables raised to the power of one, are among the most significant topics in algebra. The formula for a basic linear equation is $y = mx + c$, where m is the slope and c is the intercept. In data science, this form is frequently utilized, particularly in linear regression models. Predictions and trend identification are aided by linear equations. For instance, a linear equation can be used to predict marks based on study time if x stands for the number of hours studied and y for marks. Another essential ability is the ability to solve algebraic equations. It entails determining the variable's value that satisfies the equation. For instance, $x = 3$ is obtained by solving $2x + 4 = 10$. Equation solving is frequently used in data science to locate unknown parameters or optimize models. In order to minimize error, many machine learning algorithms modify variables, which is basically an iterative solution to algebraic equations.

Systems of equations, in which several equations are solved simultaneously, are another idea introduced by algebra. When working with several variables, this is really helpful. In data science, datasets frequently have a large number of attributes, and systems of equations can be used to model the relationships between them. For instance, a number of factors, including region, location, and number of rooms, may be involved in forecasting house prices. Algebra aids in managing the relationships between these factors, each of which contributes to the final forecast.

Inequalities, which describe relationships in which values lie within a range but are not precisely equal, are another crucial idea. For instance, $x > 10$ indicates that any value greater than 10 can be assigned to x. In data science, inequality is frequently used for data filtering, constraint setting, and decision-making. A business rule might, for example, specify that only

clients who spend more than ₹5,000 are regarded as premium clients. Inequalities can be used to express this state.

Functions, which explain how one variable depends on another, rely heavily on algebra. Based on a predetermined rule, a function generates an output from an input. For instance, the function $f(x) = 2x + 3$ has an output that varies according to the input value of x. Functions are used in data science to construct algorithms, model relationships, and transform data.

Another algebraic idea that comes up in data science is polynomials. An expression made up of variables raised to various powers, such $x^2 + 3x + 2$, is called a polynomial. Non-linear relationships in data are modeled using polynomial functions. For instance, polynomial regression can be used to fit a curve if the data does not follow a straight line. Data scientists are able to identify more intricate patterns as a result.

Although matrices are frequently studied independently, algebra and matrix operations are tightly related. In data science, matrices are used in a lot of algebraic calculations, particularly when dealing with big datasets. For example, a matrix can be used to represent data in tabular form, and algorithms use operations like addition and multiplication. Gaining an understanding of mathematics facilitates the comprehension of these complex ideas.

Algebra is utilized in domains including data transformation, optimization, and predictive modeling in real-world data science applications. For instance, the best-fitting line for a dataset is determined using algebra in linear regression. Algebra aids in defining decision limits in categorization tasks. Algebraic transformations are used to scale or standardize data even in data preprocessing. It is crucial to stress that algebra in data science is about comprehending relationships between variables rather than memorizing formulas.

1.4 Function and Graphs in Data Science

Graphs and functions are basic mathematical ideas that are essential to data science. They aid in comprehending the relationships between various variables and the effects of changes in one. The majority of data science issues entail examining the connections between inputs (features) and outputs (results), and functions offer an organized method for quantitatively expressing these connections.

A rule that gives each input exactly one output is known as a function. Typically, it is expressed as f(x), where x is the input and f(x) is the output. For instance, the function $f(x) = 2x + 3$ generates the result by multiplying the input value by 2 and then adding 3. Functions are used in data science to simulate real-world processes like stock price forecasting, temperature estimation, and sales prediction.

Functions are crucial because they facilitate the transformation of unprocessed data into insightful knowledge. Take a dataset, for instance, in which y represents exam scores and x represents the number of hours studied. The relationship between marks and study hours can be described by a function. Data scientists can anticipate outcomes and recognize trends in the data because to this link. It would be challenging to extrapolate patterns beyond observed data points in the absence of functions. In data science, a variety of functions are frequently employed. The linear function, which takes the form $y = mx + c$, is among the most basic. In this case, c stands for the intercept and m for the slope. Regression analysis makes extensive use of linear functions due to their ease of comprehension and interpretation. They work well when there is a roughly straight-line relationship between the variables, like when pay is predicted based on years of experience. The quadratic function, which has the formula $y = ax^2 + bx + c$, is another significant kind. When data has a non-linear pattern, quadratic functions are helpful because they generate curved graphs. For instance, a quadratic trend may be seen in the rise and fall of product sales over time. These functions aid in capturing more intricate relationships that are difficult for straight lines to depict. In data science, exponential functions are very frequently employed, particularly when simulating growth or decay processes. The formula for a typical exponential function is $y = a \cdot b^x$. These capabilities are helpful in areas like financial investments, disease transmission, and population expansion. Exponential functions are utilized in machine learning methods such as neural networks and logistic regression.

Graphs give data and functions a visual form. When a graph is plotted using a Cartesian coordinate system, the input variable is represented by the horizontal axis (x-axis) and the output variable by the vertical axis (y-axis). A pair of values (x, y) are represented by each point on the graph. Graphs aid in the comprehension of relationships, patterns, and trends that may not be immediately apparent from unprocessed data. The scatter plot, which shows individual data points on a graph, is one of the most widely used graphical tools in data analysis. To find relationships between variables, scatter plots are helpful. Plotting study hours against grades, for instance, can show whether the two are positively correlated. A positive correlation is shown if the points show an upward trend; a negative correlation is indicated if the points show a downward trend. The line graph, which is frequently used to display trends over time, is another crucial graphical representation. To see changes in temperature, sales, or website traffic over days, months, or years, for instance, a line graph can be utilized. Line graphs are particularly useful in time series analysis, where understanding trends and patterns over time is essential. Important features of functions, such slope, intercepts, and curvature, can also be

found using graphs. A graph's slope shows how quickly variables are changing. Whereas a mild slope denotes a slower change, a steep slope denotes a faster change. Slope is a crucial idea in regression analysis and is used in data science to quantify correlations. The idea of domain and range is another crucial component. Whereas the range denotes all potential output values, the domain denotes all potential input values. Knowing domain and range in data science aids in identifying acceptable inputs and accurately interpreting findings. Negative values, for instance, might not make sense in some situations, such when it comes to age or the quantity of goods sold.

Graphs and functions are also crucial for assessing and interpreting models. Graphs are frequently used by data scientists to show how well a model fits the data once it has been constructed. Plotting predicted values against actual values, for instance, aids in determining the model's accuracy. To examine mistakes and enhance model performance, residual plots are utilized. Furthermore, data preprocessing frequently makes use of function transformations. For instance, using scaling or logarithmic adjustments might enhance model performance and facilitate data analysis. Graphs can also be used to depict these modifications, which makes it simpler to comprehend how they affect the data.

Check your progress-2

1. _____offers a methodical technique to depict the relationships between variables
2. Variables and constants are fundamental to algebra (True/False)
3. List out any two variables which are used in student performance dataset.
4. What do you mean by polynomial?
5. A rule that gives each input exactly one output is known as a _____.
6. Graphs give data and functions a visual form (True/False)

1.5 Let us sum up

In this unit we have discussed that why mathematics is very important for Data Science. The unit also discussed the various number systems used in Data Science. The unit also discussed the basic concepts of algebra in context to Data Science. Finally the unit is ended with the concepts of functions and graphs.

1.6 Check your progress: Possible Answers

1-a

Data science is built on mathematics, which is essential to comprehending, evaluating, and interpreting data.

1-b Counting numbers beginning with 1 are known as natural numbers. For instance, 1, 2, 3, 4, 5,

1-c Ratios,Probabilities, percentages etc 1-d Statistics

1-d Binary

1-e Integers, floating-point numbers, and occasionally higher-precision formats like decimal

2-a Algebra

2-b True

2-c study hours, attendance, and grades

2-d An expression made up of variables raised to various powers, such $x^2 + 3x +$ **2-e** function

2-f True

1.7 Further Reading

1. "Essential Maths for Data Science", Thomas Nield, O'Reilly, May 2022
2. https://www.geeksforgeeks.org/data-science/linear-algebra-required-for-data-science/
3. https://neo4j.com/blog/graph-data-science/understanding-graphs-and-graph-data-science/

1.8 Assignments

1. Why mathematics is important for Data Science?
2. Explain variables and constant concepts of algebra.
3. Explain inequality concept of algebra.
4. What are the applications of algebra in Data Science?
5. Explain the role of function and graph in Data Science.

Unit-2: Descriptive Statistics

Unit Structure

2.0 Learning Objectives

After studying this unit student should be able to:

- Able to understand the importance of Descriptive Statistics in Data Science
- Able to understand Central Tendency measures such as mean,median, mode
- Able to apply Central Tendency measures in real life applications
- Able to understand Measure of Variability measures such as Range, Quantile, Inter Quantile Range, Variance, Stadard Deviation
- Able to apply Measure of Variability measures in real life applications

2.1 Introduction

Descriptive statistics are essential for comprehending, summarizing, and preparing data in data science. It prepares the ground for more in-depth analysis, which enables data scientists to create models that are more precise and efficient. The data would be overwhelming and challenging to understand without descriptive statistics.

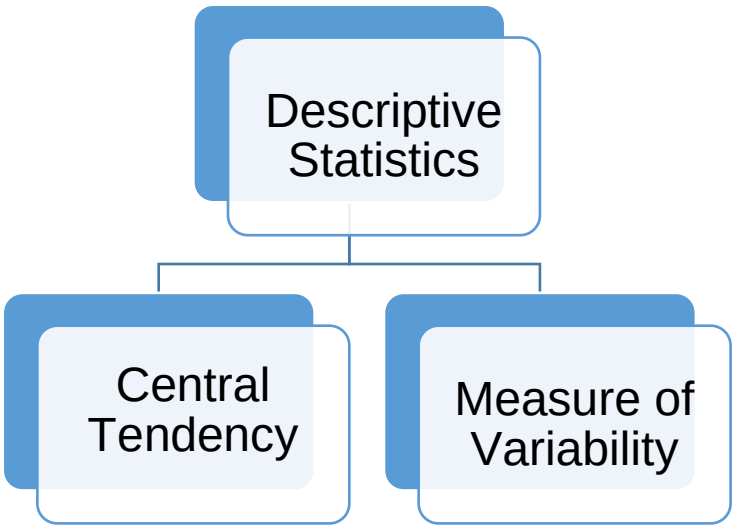
With the help of descriptive statistics, data scientists can rapidly comprehend the variability (range, variance, and standard deviation) and central tendencies (mean, median, and mode) of datasets. Prior to additional analysis, this aids in obtaining a broad picture of the data.

Data visualization methods like bar charts, box plots, and histograms are based on descriptive statistics. Better insights are provided by these visuals, which make it simpler to spot trends, patterns, and outliers in the data.

In order to guarantee data quality, descriptive statistics help identify outliers, missing values, and inconsistencies in the data. For example, finding extreme values using metrics like standard deviation or interquartile range is crucial for accurate modelling because these can distort results.

Charts, graphs, tables, Excel files, and other representative techniques are used in descriptive statistics to describe our data. Descriptive statistics involves describing our data in some way and presenting it in a way that makes sense so that it is easy to understand. It is typically conducted on small data sets, and based on the current findings, this analysis greatly aids in forecasting some future trends. Measures of central tendency, variability, and dispersion are a few metrics used to characterize a data set.

Descriptive statistics categories mainly in two types as described in Figure 2.1.1.



(Figure 2.1.1 Categories of Descriptive Statistics)

Central Tendency uses a single value to represent the entire set of data. It provides us with the central points' locations. Measures of variability are also known as measures of dispersion because they provide information about the spread or dispersion of the current observations.

2.2 Central Tendency

A statistical metric known as central tendency designates a single value as representative of a whole data distribution. By emphasizing a dataset's "typical" value, it serves as a summary or focal point and provides insights into the nature of the data. The mean, median, and mode are the three primary indicators of central tendency.

2.2.1 Arithmetic Mean

Arithmetic mean of observations is the sum of observed values in the data divided by total number of observations. It is represented as equation-1.

$$A=1\ Ni=1nAi \qquad \text{----------(1)}$$

Where A is Arithmetic Mean

Ai is each value of observations such as A1, A2, A3… AN

N is total number of observations

Example-1: Find Arithmetic mean of marks of following data:

63,62,52,54,82,43,81,92,38,64

By using the formula for calculating the Arithmetic Mean or the average, we take:

(63+62+52+54+82+43+81+92+38+64) / 10 = 631 /10 = 63.1

Example-2: Find Arithmetic mean of Students Attendance in this week

75,78,72,80,82,65

Arithmetic mean = (75+78+72+80+82+65) = 452/6 = 75.33

Advantages:

It is easy to calculate. The formula is very simple.

It is comprehensive as uses all the observations equally.

It applicable in various real time applications.

Disadvantages:

It is sensitive to outliers. It can be distorted by extreme values.

It is not possible in real life that all observations always provide equal contribution to final mean. Some observations may contribute more than others.

2.2.2 Weighted Mean

It is one form of Arithmetic Mean where certain data points add more "weight" to the final mean than others, rather than each one contributing equally. The weighted mean is equal to the arithmetic mean if all the weights are equal. The weighted mean can be found by:

1. Multiply the weights by the observations in your data set.
2. Total the outcomes.
3. Subtract it from the total weights.

It is represented by equation-2.

$A = i=1NAi*WiWi$ ---------- (2)

Where A is Weighted Mean

Ai is each value of observations such as A1, A2, A3… AN

Wi is weights associated with each observations such as W1,W2,W3 …WN

Example-1: Assume that the results of students in any course is depicted as following table

No.of Students	Marks
2	70
4	80
5	50
3	65
10	40
1	95

Determine the average marks of students.

Here, observations are marks of students. All observations are not contributing equally to final mean so Weighted mean is the right technique to calculate the average marks of students. Here, marks are the observations. Total number of students are considered as weights.

$$\text{Average Marks} = \frac{2*70 + 4*80 + 5*50 + 3*65 + 10*40 + 1*95}{2 + 4 + 5 + 3 + 10 + 1}$$

$$= 1400/25 = 56$$

Example-2: You take three 100-point exams in your statistics class and score 80, 80 and 95. The last exam is much easier than the first two, so your professor has given it less weight.

The weights for the three exams are:

Exam 1: 40 % of your grade.

Exam 2: 40 % of your grade.

Exam 3: 20 % of your grade.

What is your final average of the marks?

Here, observations are marks of students. All observations are not contributing equally to final mean so Weighted mean is the right technique to calculate the average marks of students. Here, marks are the observations. Weightage of tests are considered as weights.

$$\text{Average Marks} = (40*80 + 40*80 + 20*95)/100$$

$$= (3200 + 3200 + 1900)/100$$

$$= 83$$

Advantages:

it ideal for situations where not all data points should contribute equally to the overall result.

It is adaptable as it can be used in a variety of situations by changing the weights.

Disadvantages:

Because it necessitates knowledge of both the data points and their corresponding weights, the weighted mean calculation is more complicated than the arithmetic mean.

Sensitive to outlier

The weighted mean is predicated on the accuracy of the assigned weights and the data points.

2.2.3 Median

It is the observation set's most middle value. The order of the observations must be ascending. One middle value will be obtained if the total number of observations is odd; if not, two values will be obtained, and we must average the two values.

Example-1: Calculate the median value of following observations:

5,7,2,3,1,4,8

Ascending order for the same is: 1,2,3,4,5,7,8

Here total number of observations are odd so only one middle value will be get. Here total number of observations are 7 so 4^{th} value is the middle value.

So in the above example the median value is 4 (4^{th} values in observation)

Example-2: Calculate the median value of following observations:

1,2,3,5,7,2,4,1,4,8

Ascending order for the same is: 1,1,2,2,3,4,4,5,7,8

Here total number of observations are even so two middle values will be get. Here total number of observations are 10 so 5^{th} and 6^{th} values are middle values.

In above example, 3 and 4 values are middle values

Median = (3 + 4) /2 = 3.5

Advantages:

It is robust to extreme values and relies on the middle value or values.

Improved depiction of central tendency in skewed data.

Disadvantages:

The median ignores the rest of the dataset and only takes into account the middle value or values, in contrast to the mean.

In more complex statistical methods, the mean is more valuable than the median.

2.2.4 Mode

It is the data set's most common occurrence.

1. Dataset has no mode if its greatest frequency is 1 (no value appears more than once).

2. Any value that occurs with the highest frequency, if it is 2 or higher, is referred to as the variable's mode.

Example-1: Find the mode of following observations:

1,2,3,5,7,2,3,1,4,8,1

Here, in the example, 1 occurs more times so mode is 1.

Example-2: Find the mode of following observations:

1,2,2,1,3,4,5,2,1,3,2,1

Here, in the example, 1 and 2 occurs highest times so mode of the data is: 1 and 2

Example-3: Find the mode of following observations:

1,5,3,2,6,9,7,8

Here, in the example, all elements occur only once so there is no mode of the data,

Advantages:

It is simple to recognize and understand the mode.

Useful for nominal data

Disadvantages:

Multiple modes in a dataset can complicate interpretation.

In datasets with evenly distributed values or continuous data, the mode might not be a reliable indicator of the central tendency.

2.2.5 How to choose appropriate measure of Central Tendency?

In Data Science project, the selection of appropriate Central Tendency measure is vital. There are number of aspects which should be considered while selecting an appropriate measure of Central Tendency.

Data Type: Data Type is main aspect for selection of right Central Tendency value. The mode is appropriate for nominal data, the median is appropriate for ordinal, interval, and ratio data, and the mean is appropriate for interval and ratio data.

Distribution Shape of Data: The shape of distribution of data is another important aspect for the selection of appropriate Central Tendency measures. For symmetric, normally distributed data, the mean works best. The mode is helpful when working with categorical data or when a dataset contains multiple peaks, while the median is better for skewed distributions.

Outlier Condition: The median is typically a more accurate indicator of central tendency than the mean when a dataset contains outliers.

Check Your Progress-1

1. What are the types of Descriptive Statistics?
2. What is the objective of Descriptive Statistics?
3. A statistical metric known as central tendency designates a _____ value as representative of a whole data distribution.
4. Find the arithmetic mean of following observations:
   6,8,4,3,5,4,3,8,7,6,5
5. Calculate median of following observations
   1,3,2,1,3,4,5,2,4,6,7

2.3 Measure of Variability

Measure of Variability is one kind of statistical metric that measures how much a dataset's data points deviate from one another or from a central value (such as the mean, median or mode) is known as a measure of variability, or measure of dispersion.

A crucial component of descriptive statistics is variability. The mean, median, and mode are examples of measures of central tendency that provide information about the typical or central value in a dataset, but they do not reveal the degree of dispersion of the data points. It is possible for two datasets to have completely different spreads but the same mean. The range, variance, standard deviation and Quantile are primary indicators of measure of variability.

2.3.1 Range

The easiest measure of variability to compute is the range, which you have undoubtedly seen many times in your life. It is the variation between a data set's maximum and minimum values. The equation-3 depicts the formula of Range.

Range = Maximum - Minimum ----------(3)

Where Maximum is the maximum value from the dataset

Minimum is the minimum value from the dataset

Example-1: The score of the first unit test of Data Science course is as under:

1,5,4,3,2,7,8,9,10,2,11,12,5,7,8,9,10

Calculate the range for the same.

Here, maximum value of observations is 12. The minimum value is 1.

Range = 12 -1 =11

Example-2: The temperature of Ahmedabad in January month is as under:

17,20,19,21,19,18,25,22,21,20,16,15

Calculate the range for the same.

Here, maximum value of observations is 22. The minimum value is 15.

Range = 22 -15 =7

2.3.2 Quartile or Quantile

The word quantity is the root of the word "quantile." To put it simply, a quantile is a sample that has been split up into multiple subgroups. An observation data set has multiple quartiles. When data is sorted in ascending order, the value that removes the first 25% of the data is known as the first quartile, or lower quartile. The value that separates the first 50% is known

as the median or second quartile. The value that separates the first 75% is known as the third quartile, or upper quartile. There are various algorithms exist in literature to find Quartile or Quantile. Here, the most popular algorithm is depicted to determine Quartile or Quantile.

v = sort(v)

h = ((length(v)-1)*p)+1

v[floor(h)]+((h-floor(h))*(v[floor(h)+1]- v[floor(h)]))

where v is the vector that contains the data items.

p is the probability of the quartile.

length is the function which determines the length of the vector.

floor is the function that determines the floor value of a value.

Example-1: Assume that we have a data set of marks of students as under :

25,20,30,40,45,50,60,70,80,35,90,95

Determine the 25% quartile based on the above data.

v= 20,25,30,35,40,45,50,60,70,80,90,95

h= (11 *0.25) +1 = 3.75

v[floor(h)] =v[3]= 30

h – floor(h)= 3.75 – 3 = 0.75

v[floor(h)+1] =v[4]= 40

so v[floor(h)]+((h-floor(h))*(v[floor(h)+1]- v[floor(h)])) = v[3] + ((3.75-3) * (v[4]-v[3]))

= 30 + ((0.75) *5)

= 33.75

Example-2: Determine the 50% quartile based on the above data of Example-1.

v= 20,25,30,35,40,45,50,60,70,80,90,95

h= (11 *0.50) +1 = 6.5

v[floor(h)] = v[6]=45

h – floor(h)= 6.5 – 6 = 0.5

v[floor(h)+1]=v[7] = 50

so v[floor(h)]+((h-floor(h))*(v[floor(h)+1]- v[floor(h)])) = v[6] + ((0.5) * (v[7]-v[6]))

= 45+ ((0.5) *5)

= 47.5

2.3.3 Inter Quantile Range(IQR)

It is the difference between third and first quartile.

IQR= Q3 –Q1 Where Q is quartile.

Example

Assume that we have a data set of marks of students as under:

25,20,30,40,45,50,60,70,80,35,90,95

Here, The first quartile value means 25% is =33.75

The third quartile value means 75% =72.50

Therefore, IQR= 72.50 -33.75 = 38.75

2.3.4 Variance

A numerical indicator of how the data values are distributed around the mean is the variance. It is the mean squared deviation from the mean. The variance of population is calculated using Equation-4

2=1N i=1N(Ai-μ)2 ----------(4)

Where 2 is Variance

Ai is each value of observations such as A1, A2, A3… AN

N is total number of observations

μ is the mean of population

Example : Determine the variance of following Observations:

10, 12,9,8,10,11,9,11,12

Here, the mean of observations is = (10+12 +9+8+10+11+9+11+12)/ 9

= 10.22222

Now calculate the variance. To determine the variance we need Ai-μ and (Ai-μ)2. After that we have to do the summation of all values of (Ai-μ)2 and then we need to divide by total number of observations. The following table contains the calculation part.

Ai	μ	(Ai- μ)	(Ai-μ)2
10	10.22222	-0.22222	0.04938173
12	10.22222	1.77778	3.160502
9	10.22222	-1.22222	1.493822
8	10.22222	-2.22222	4.938262
10	10.22222	-0.22222	0.04938173
11	10.22222	0.77778	0.6049417
9	10.22222	-1.22222	1.493822
11	10.22222	0.77778	0.6049417
12	10.22222	1.77778	3.160502
		Summation	15.55556

the variance of population = 15.55556/9

= 1.728395

2.3.5 Standard Deviation

The square root of variance is considered as standard deviation. The standard deviation is depicted in equation-5.

σ=1N i=1N(Ai-μ)2 ----------(5)

Example: 10, 12,9,8,10,11,9,11,12

Here, the mean of observations is = (10+12 +9+8+10+11+9+11+12)/ 9

= 10.22222

Now calculate the variance. To determine the variance we need Ai-μ and (Ai-μ)2. After that we have to do the summation of all values of (Ai-μ)2 and then we need to divide by total number of observations. After that take square root of it. The following table contains the calculation part.

Ai	μ	(Ai- μ)	(Ai-μ)2
10	10.22222	-0.22222	0.04938173
12	10.22222	1.77778	3.160502
9	10.22222	-1.22222	1.493822
8	10.22222	-2.22222	4.938262
10	10.22222	-0.22222	0.04938173
11	10.22222	0.77778	0.6049417
9	10.22222	-1.22222	1.493822
11	10.22222	0.77778	0.6049417
12	10.22222	1.77778	3.160502
		Summation	15.55556

the variance of population = 15.55556/9

= 1.728395

The standard deviation is =Variance = 1.728395 = 1.31468

2.3.6 Importance of Measure of Variability

Measure of Variability has number of advantages. It is depicted as follows:

Making Decisions: In the business world, understanding the fluctuations in demand, sales, or other important metrics can aid in risk management and decision-making.

Risk assessment: The standard deviation, a measure of variability, is used in finance to evaluate the risk of investments. A more volatile investment is indicated by a higher standard deviation.

Quality Control: Maintaining constant quality and reducing flaws in manufacturing is made easier by an awareness of process variability.

Investigations and Scientific Research: Variability in experiments aids in evaluating the consistency and dependability of findings.

Check your progress-2

1. The score of the first unit test of Data Science course is as under:
    a. 10,11,5,3,5,20,22,12,11,10,6,8
    b. Calculate the range for the same.
2. The square root of variance is considered as _____.
3. Determine the variance of following Observations:
   6,3,5,7,8,3,4,9,3,8

2.4 Let us sum up

In this unit, overview of Descriptive Statistics with types were discussed. The measures of Central Tendency such as Mean, Median and Mode were discussed with examples and advantages and disadvantages. The unit also discussed the objective and measures of Measure of Variability with examples.

2.5 Check your progress: Possible Answers

1-a Central Tendency and Measure of Variablity

1-b Descriptive statistics are essential for comprehending, summarizing, and preparing data in data science

1-c Single

1-d (6+8 4+3+5+4+3+8+7+6+5) = 59/11 = 5.36

1-e Ascending Order: 1,1,2,2,3,3,4,4,5,6,7

Median =3

2-a Maximum= 22, Minimum=3, Range = 22 -3 = 19

2-b Standard Deviation

2-c Variance = 4.8399

2.6 Further Reading

1. https://researchmethod.net/descriptive-statistics/pes, Methods and Examples
2. Julie Scott Jones , John Goldring , " Exploratory and Descriptive Statistics",Sage Publication,2021.

2.7 Assignments

1. What is objective of Descriptive Statistics?
2. Find Arithmetic Mean of following observations:

   43,45,67,42,78,32,65,61,64
3. Assume that the temperature in any days is depicted as following table:

No.of days	Temperature
4	42
3	35
4	38
3	36
2	40
3	41

Find average of temperature.

4. Find the median of following data:

   32,30,31,34,38,37,35,31,30
5. Find the mode of following data:

   7,5,4,5,4,6,7,3,4,5
6. Find the range of following marks data :

   12,11,10,9,10,11,12,13,12,11,10
7. Determine the variance of following Observations:

   11, 10,11,8,8,10,11,9,10,11
8. Find the standard deviation of the data mentioned in Question-7.

Unit-3: Probability Theory

Unit Structure

3.0 Learning Objectives

After studying this unit student should be able to:

- Understand basic concepts and rules of probability
- Able to understand simple and classical probability with various categories of examples.
- Able to understand Conditional Probability with examples

3.1 Introduction

A subfield of mathematics known as probability theory studies the possibility or chance that events will occur. It offers a structured framework for expressing uncertainty, which is essential in many disciplines, especially data science. The basis for comprehending randomness, forecasting results, and making data-driven decisions is provided by probability theory. Numerous data science procedures, including the creation of machine learning algorithms and the examination of real-world datasets, are based on probability theory.

Probability is important in data science for tasks like:

Formulating Predictions: Based on data, probability is utilized to model uncertain outcomes and formulate predictions.

Inference: It facilitates the extrapolation of patterns from samples to broader populations.

Modelling Randomness: It makes it possible to represent the intrinsic unpredictability of data, which is crucial for comprehending variations.

In order to understand probability effective, you should learn some basic concepts associated with it.

Experiment is the process or any action that generates well-defined outcomes.

Sample Space is one of the basic concept associated with probability. The set of all possible outcomes of a random experiment is known as sample space. For example, if three coins are tossed independently then the sample space of this event is { HHH,HHT,HTH, HTT, THH,TTH,THT,TTT}.

Event is subset of sample space. Any event consists of one or more outcomes. In above example of sample space, the event of getting exactly two heads, or at least one head, at least one tail are various examples of event.

Two events are **mutually exclusive** if they cannot occur at the same time. For example, flipping a coin, getting head and tail are mutually exclusive events.

If the occurrence of one event has no bearing on the occurrence of the other, then the two events are **independent.** For example, rolling two dice, the results of first die does not impact the result of second die.

The probability theory has certain rules and regulations.

The probability of any event is $0 <= P(E) <= 1$ where 0 means the event will never occurs and 1 means always occurs.

The probability that an event does not occur is 1 minus the probability that it does occur. It is known as complimentary rule.

$$P(E^c) = 1 - P(E) \quad \text{------ (8)}$$

For example, if probability of wind storm is 0.40 then, the probability of not having wind storm is

P(Not having wind storm) = 1 – 0.4 = 0.6

For mutually exclusive events, the probability of either event A or event B occurring is the sum of their individual probabilities.

$$P(A \cup B) = P(A) + P(B) \quad \text{------(9)}$$

Where P(AUB) is the union of two events A and B

P(A) is the probability of event A

P(B) is the probability of event B

For Example, Let's say a fair six-sided die is rolled. Let's assume that you roll a 3 in event A and a 4 in event B. These events are mutually exclusive because they cannot occur simultaneously

P(A)= 1/6 and P (B) = 1/6

P(AUB) = 1/6 + 1/6 = 2/6 =1/3

So the probability of rolling either 3 or 4 is 1/3

The occurrence of one event has no bearing on the occurrence of the other when they are independent. We can determine the likelihood that both events will occur by applying the Multiplication Rule for independent events. According to the rule, the likelihood that two independent events, AA and BB, will occur together (i.e., that A and B will intersect) is equal to the product of their respective probabilities:

$$P(A \cap B) = P(A) * P(B) \quad \text{------(10)}$$

Where $P(A \cap B)$ is the probability that both event A and B occur

P(A) is the probability of event A

P(B) is the probability of event B

Let's say you flip two coins. Assume that the first coin will come up heads in event A, and the second coin will come up heads in event B. These events are independent since the outcome of the first coin toss has no bearing on the outcome of the second.

P(A)= 1/2 and P (B) = 1/2

P(AUB) = 1/2 * 1/2 = 1/4 =1/4

So the probability is 1/4

3.2 Simple or Classical Probability

The foundation of classical probability, sometimes referred to as simple probability, is the idea that every possible outcome in an experiment is equally likely. When every possible outcome has an equal chance of happening, it is used to determine the probability of an event.

The formula of Simple or Classical probability is depicted in equation-11.

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes) –-------(11)

Where P(E) represents the simple probability of any event E

Favourable outcomes are outcomes that satisfy any event E

3.2.1 Flipping a coin examples of Classical Probability

Example 1: Calculate the probability of getting a head, when coin is flipped.

In this example, when a coin is flipped, there are two possibilities of it i.e Head or Tail.

So the sample space is = { H,T} where H= Head and T=Tail.

Means total number of **possible outcomes are 2**.

Here, we are interested to get the probability of having a head. Here we have only one favourable outcome that satisfy the condition.

So the number of **favourable outcomes is 1.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 1/ 2 = 0.5

Example 2: A fair coin is flipped twice. Determine the probability of following situations:

a. Getting at least one Head

b. Getting at most one Head

c. Getting exactly two Tails

d. Getting no Tails

(a)Getting at least one Head: Here Sample space means total number of possible outcomes are = { HH, HT, TH,TT}

Number of favourable outcomes having at least one head is = {HH,HT,TH}

So, P(E) = 3/ 4 = 0.75

(b) Getting at most one Head: Here Sample space means total number of possible outcomes are = { HH, HT, TH,TT}

Number of favourable outcomes having at most one head is = { HT,TH,TT}

So, P(E) = 3/ 4 = 0.75

(c) Getting exactly two Tails : Here Sample space means total number of possible outcomes are = { HH, HT, TH,TT}

Number of favourable outcomes having exactly two tails is = { TT}

So, P(E) = 1/ 4 = 0.25

(d) Getting no Tails : Here Sample space means total number of possible outcomes are = { HH, HT, TH,TT}

Number of favourable outcomes having no tail is = { HH}

So, P(E) = 1/ 4 = 0.25

Example 3: A fair coin is flipped three times. Determine the probability of following situations:

a. Getting all Heads
b. Getting more Heads Than Tails
c. Getting at least two Tails
d. Getting even number of Tails

(a)Getting all Heads: Here Sample space means total number of possible outcomes are = { HHH, HHT, HTH,HTT,TTT,TTH,THT,THH}

Number of favourable outcomes having all Heads = {HHH}

So, P(E) = 1/ 8 = 0.125

(b) Getting more Heads Than Tails : Here Sample space means total number of possible outcomes are = { HHH, HHT, HTH,HTT,TTT,TTH,THT,THH}

Number of favourable outcomes having more Heads than Tails= {HHH,HHT,HTH,THH}

So, P(E) = 4/ 8 =0.5

(c) Getting at least two Tails : Here Sample space means total number of possible outcomes are = { HHH, HHT, HTH,HTT,TTT,TTH,THT,THH}

Number of favourable outcomes having at least two Tails= { HTT,TTT,TTH,THT }

So, P(E) = 4/ 8 =0.5

(d) Getting even number of Tails : Here Sample space means total number of possible outcomes are = { HHH, HHT, HTH,HTT,TTT,TTH,THT,THH}

Number of favourable outcomes having even number of tails= {HTT,TTH,THT }

So, P(E) = 3/ 8 =0.375

3.2.2 Rolling a die examples of Classical Probability

Example 1: Calculate the probability of getting 4, when six sided fair die is rolled.

In this example, when a six sided fair die is rolled, there are six possibilities of it i.e 1,2,3,4,5,6

So the sample space is = { 1,2,3,4,5,6}

Means total number of **possible outcomes are 6**.

Here, we are interested to get the probability of having 4. Here we have only one favourable outcome that satisfy the condition.

So the number of **favourable outcomes is 1.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 1/ 6 = 0.166

Example 2: Calculate the probability of getting even numbers, when six sided fair die is rolled.

In this example, when a six sided fair die is rolled, there are six possibilities of it i.e 1,2,3,4,5,6

So the sample space is = { 1,2,3,4,5,6}

Means total number of **possible outcomes are 6**.

Here, we are interested to get the probability of getting even numbers. Here we have three favourable outcome that satisfy the condition.

So the number of **favourable outcomes is 3.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 3/ 6 = 0.50

Example 3: Calculate the probability of getting prime number, when six sided fair die is rolled.

In this example, when a six sided fair die is rolled, there are six possibilities of it i.e 1,2,3,4,5,6

So the sample space is = { 1,2,3,4,5,6}

Means total number of **possible outcomes are 6**.

Here, we are interested to get the probability of getting prime numbers. Here we have three favourable outcome that satisfy the condition.

So the number of **favourable outcomes is 3.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 3/ 6 = 0.50

Example 4: Calculate the probability that sum of outcomes is greater than 10, when Two six sided fair dice are rolled.

Here, when two six sided fair dice are rolled total 36 outcomes achieved as depicted in the below table.

	1	2	3	4	5	6
1	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)
2	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)
3	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)
4	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)
5	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)
6	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)

In this example, when tow six sided fair die is rolled, there are thirty-six possibilities of it. Means total number of **possible outcomes are 36**.

Here, we are interested to get the probability that sum of outcomes is greater than 10. Here we have three favourable outcome that satisfy the condition i.e (5,6),(6,5),(6,6)

So the number of **favourable outcomes is 3.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 3/ 36 = 0.083

Example 5: Calculate the probability of getting one odd and one even number, when Two six sided fair dice are rolled.

In this example, when tow six sided fair die is rolled, there are thirty-six possibilities of it. Means total number of **possible outcomes are 36**.

Here, we are interested to get the probability of getting one odd and one even number. The number of transactions are = {(1,2),(1,4),(1,6),(2,1),(2,3),(2,5),(3,2),(3,4),(3,6),(4,1),(4,3),(4,5),(5,2),(5,4),(5,6),(6,1),(6,3),(6,5) }

So the number of **favourable outcomes is 18.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 18/ 36 = ½=0.5

3.2.3 Important Features of Classical Probability

Equally Likely Outcomes: According to classical probability, all outcomes have an equal chance.

Finite Sample Space: This type of analysis usually focuses on experiments with a limited number of potential outcomes.

Fixed Formula: The ratio of favourable outcomes to possible outcomes is used in this simple calculation.

Check Your Progress-1

1. Give various applications of probability in Data Science.
2. A fair coin is flipped twice. Determine the probability of getting at most one Tail.
3. Flipping a coin, getting head and tail are _____ events. Mutually Exclusive/ Independent)
4. Calculate the probability of getting both odd numbers, when Two six sided fair dice are rolled.

3.3 Conditional Probability

The probability of an event happening given that another event has already happened is known as conditional probability. It measures the change in the probability of one event when we are aware that another has taken place. Given that event B has taken place, the likelihood of event A is expressed as P(A/B). It is referred as probability of A given probability of B. Equation-12 represents the formula of the conditional probability.

$P(A/B) = P(A \cap B)/P(B)$ -------(12)

Where $P(A \cap B)$ is the probability that both events A and B occur.

P(B) is the probability that event B occurs

Here, P(B) != 0 as we can't condition on an event that has zero probability.

We use conditional probability frequently in our daily lives, despite the fact that it may appear to be a complex idea.

In order to predict the probability of future weather conditions based on current conditions, weather forecasters employ conditional probability. If it's cloudy outside, they might figure out how likely it is to rain.

To determine the odds that specific teams will win their match, sports betting companies may employ conditional probability. These odds might depend on information about the team, like being aware of a key player's injury

When a doctor and patient discuss a vaccine's efficacy, they may employ conditional probability. To put it another way, a patient who receives the vaccine may have a significantly lower risk of catching a virus than one who does not.

Conditional probability is used by insurance companies to determine auto insurance rates. For instance, if you are younger than 20, the insurance company might assume that you have a higher risk of getting into an accident.

Example-1: Two fair dice are rolled and the sum of the numbers is observed. What is the probability that the sum is at least 9 if it is known that a 5 was rolled?

Here, we need to find the probability that sum of two rolled dice is at least 9 if it is known that a 5 was rolled means P(Sum is at least 9 / 5 was rolled)

P(A=Sum is at least 9 if it is known that 5 was rolled) = { (4,5),(5,4),(5,5),(5,6),(6,5)} = 5

P(5 was rolled) ={(1,5),(2,5),(3,5),(4,5),(5,5),(6,5),(5,1),(5,2),(5,3),(5,4),(5,6)} = 11

P(Sum is at least 9 / 5 was rolled) = 5/11 = 0.45

Example-2: For a group of people, the probability of having blond hair is 25%. The probability of having blond hair and blue eyes is 10%. What is the probability that person has blue eyes given they have blond hair?

Here, P(A) = P(blond hair) = 0.25

P(A and B)=P(blond hair and blue eyes) = 0.10

P(A and B) / P(A)= (0.10) / (0.25)= 2/5 = 0.40

Example-3: We roll a six-sided die. What is the probability that the roll is a 6, given that the outcome is an even number.

Here, P(A and B)/ P(B) = P(Roll is 6/ Outcome is even number) = 1/3

Check Your Progress-2

1. Give any two example of daily life where conditional probability can apply.
2. Two fair dice are rolled and the sum of the numbers is observed. What is the probability that the sum is at least 7 if it is known that a 3 was rolled?

3.4 Let us sum up

In this unit we have discussed overview of Probability Theory. The unit started with basics of probability with necessary concepts. Unit discussed the classical or simple probability with various examples. At, last the unit discussed the special types of probability that is conditional probability with appropriate examples.

3.5 Check your progress: Possible Answers

1-a Applications are

Formulating Predictions: Based on data, probability is utilized to model uncertain outcomes and formulate predictions.

Inference: It facilitates the extrapolation of patterns from samples to broader populations.

Modelling Randomness: It makes it possible to represent the intrinsic unpredictability of data, which is crucial for comprehending variations.

In order to understand probability effective, you should learn some basic concepts associated with it.

1-b Getting at most one Tail: Here Sample space means total number of possible outcomes are = { HH, HT, TH,TT}

Number of favourable outcomes having at most one tail is = { HT,TH,HH}

So, P(E) = 3/ 4 = 0.75

1-c Mutually Exclusive

1-d

In this example, when tow six sided fair die is rolled, there are thirty-six possibilities of it. Means total number of **possible outcomes are 36**.

Here, we are interested to get the probability of getting both odd numbers. The number of transactions are = {(1,1),(1,3),(1,5),(3,1),(3,3),(3,5), (5,1),(5,3),(5,5)) }

So the number of **favourable outcomes is 9.**

P(E) = (Number of Favourable Outcomes) / (Total number of Possible Outcomes)

= 9/ 36 = 1/4=0.25

2-a

In order to predict the probability of future weather conditions based on current conditions, weather forecasters employ conditional probability. If it's cloudy outside, they might figure out how likely it is to rain.

To determine the odds that specific teams will win their match, sports betting companies may employ conditional probability. These odds might depend on information about the team, like being aware of a key player's injury

2-b

Two fair dice are rolled and the sum of the numbers is observed. What is the probability that the sum is at least 7 if it is known that a 3 was rolled?

Here, we need to find the probability that sum of two rolled dice is at least 7 if it is known that a 5 was rolled means P(Sum is at least 7 / 3 was rolled)

P(A=Sum is at least 7 if it is known that 3 was rolled) = { (3,4),(3,5),(3,6),(4,3),(5,3),(6,3)} = 6

P(3 was rolled) ={(1,3),(2,3),(3,3),(4,3),(5,3),(6,3),(3,1),(3,2),(3,3),(3,4),(3,6)} = 11

P(Sum is at least 7 / 3 was rolled) = 6/11 = 0.5454

3.6 Further Reading

1. https://services.math.duke.edu/~rtd/PTE/PTE5_011119.pdf
2. Achim Klenke, " Probability Theory" - A Comprehensive Course, Springer,2020

3.7 Assignments

1. 1. Explain rules of probability theory.
2. A fair coin is flipped twice. Determine the probability of following situations:
    - Getting at least one Head
    - Getting at most one Head
    - Getting exactly two Tails
    - Getting no Tails
3. Calculate the probability of getting one odd and one even number, when Two six sided fair dice are rolled.
4. Two fair dice are rolled and the sum of the numbers is observed. What is the
5. Probability that the sum is at least 9 if it is known that a 5 was rolled?
6. Give any four examples of conditional probability that can be used in daily life.

Unit-4: Data Visualization and Interpretation

Unit Structure

4.0 Learning Objectives

After studying this unit student should be able to:

- Understand importance of Data Visualization in real world
- Understand various categories of visualization with appropriate examples
- Aware about tools and libraries for Visualization
- Understand basics of Data Interpretation

4.1 Introduction to Data Visualization

The process of displaying data in a graphical or visual manner, such as charts, graphs, and maps, is known as data visualization. Visualization makes it easier for us to see patterns, trends, and relationships rather than staring at big tables of statistics. Because raw data is frequently complex and challenging to analyse, visualization is essential in the discipline of data science. Data visualization facilitates quicker and more efficient decision-making. For instance, a line chart can quickly indicate whether sales are rising or falling in p cxz lace of examining a table of monthly sales.

Data Visualization is important for number of reasons:

makes difficult stuff simple

aids in recognizing trends and patterns

aids in making better decisions

clearly conveys knowledge to others.

finds data mistakes or anomalies

Depending on the type of data and the analysis's goal, there are several categories of data visualization. Each kind of visualization aids in comprehending particular elements including composition, relationships, distribution, and comparison. Table-4.1.1 describes the categories of visualizations and examples.

Table 4.1.1 Categories of Graph with examples

Categories of Visualization	**Example**
Comparison	Bar,Column
Trend	Line,Area
Distribution	Histogram, Boxplot
Relationship	Scatter,Bubble
Composition	Pie, Staked Bar
Hierarchy	Tree Maaps

Geographical	Map

One of the most popular ways to depict data is through comparison-based visualizations. By comparing values across various categories, these visualizations make it simple to spot variances and similarities. The bar chart, which uses rectangular bars to depict data values, is a common example of this genre. Each bar's height or length reveals the size of the value it represents. For example, a bar chart offers a quick and easy visual comparison when a teacher needs to compare students' grades in several disciplines. Comparing monthly or annual performance is another common usage for column charts, which are just vertical bar charts, in business reporting.

The main goal of trend-based visualizations is to illustrate how data evolves over time. These are particularly crucial for examining time-series data in order to spot trends like growth, decline, or variations. The line chart is the most popular visualization in this category. A line chart shows the direction and rate of change over time by connecting data points with lines. For instance, a business may monitor its sales success over a period of months or years using a line chart. This aids in spotting long-term trends or seasonal patterns. The area chart, which resembles a line chart but adds color to the area beneath the line, is another significant visual aid in this category.

To comprehend how data is distributed across various values, distribution-based visualizations are utilized. Because they shed light on the shape, center, and variability of the data, these visualizations are especially helpful in statistical analysis. One well-known example of this kind is a histogram. It illustrates the frequency of data points within each bin after grouping data into intervals, or bins. This makes it possible for analysts to determine if the data is skewed, normally distributed, or contains outliers. A histogram of students' grades, for example, can show if the scores are widely distributed or if the majority of students fell into a specific range. The box plot, which summarizes data using five essential values—minimum, maximum, median, and the first and third quartiles.

The goal of relationship-based visualizations is to illustrate the relationships between two or more variables. Finding potential relationships and patterns between variables requires the use of these visualizations. One of the most popular tools in this category is the scatter plot. It depicts data points on a two-dimensional plane, with each point representing a pair of values. A scatter plot, for instance, can be used to look at the connection between study time and test results. Higher study hours are linked to higher scores if the points show an upward-sloping

trend. Bubble charts can be used to simultaneously depict three variables in more complicated situations by adding the bubble's size as an additional dimension.

Visualizations based on composition are used to illustrate how several components work together to form a whole. These are especially helpful when trying to comprehend percentages or amounts. A typical example is a pie chart, in which a circle is divided into slices, each of which represents the contribution of a category to the total. A pie chart, for example, can be used to illustrate how a student divides their time between several disciplines. Pie charts are simple to comprehend, but they work best with datasets that have few categories. A more contemporary and eye-catching option are donut charts, which are pie charts with a hollow core. The stacked bar chart, which combines composition and comparison by displaying how individual components contribute to the total across many categories, is another helpful representation in this area.

Data with a tree-like or nested structure is represented via hierarchical visualizations. When working with data that can be broken down into smaller groups, these visualizations are quite helpful. Nested rectangles are used in treemaps, a common hierarchical visualization, to depict the many levels of the hierarchy. Users can rapidly identify the most important categories because each rectangle's size matches the value it conveys. For instance, a computer system's file sizes can be seen using a treemap, where larger rectangles stand in for larger files or directories. Another illustration is the sunburst chart, which provides hierarchical data in a circular arrangement and facilitates the visual comprehension of intricate structures.

Data linked to certain locations is represented using geographic visualizations. In disciplines like geography, urban planning, and business analytics, these visualizations are crucial. Choropleth maps, in which various regions are colored or shaded according to data values, are among the most popular kinds. For instance, a choropleth map can be used to display the population density of a nation's several states. The heat map, which employs color gradients to depict the intensity of data values, is another crucial visual aid. In web analytics, heat maps are frequently used to monitor user activity on a webpage, including clicks and mouse movements. An easy method to comprehend geographical variations and spatial patterns is through these visuals.

There are a number of sophisticated visualizations that are employed for specific tasks in addition to these fundamental kinds. For example, relationships between things, such social networks or communication systems, are represented by network graphs. Radar charts, sometimes referred to as spider charts, are helpful for performance analysis since they show

multivariate data in a circular manner. In project management, Gantt charts are frequently used to show the chronology of tasks and activities.

4.2 Importance of Data Visualization in Data Science

One of the most crucial elements of data science is data visualization since it serves as a link between unprocessed data and insightful conclusions. Organizations now gather vast volumes of data from a variety of sources, including online platforms, social media, business transactions, and sensors. However, raw data is frequently complicated, unstructured, and challenging to understand in its natural form. In order to make this data easier to comprehend and evaluate, data visualization converts it into graphical formats like charts, graphs, and maps. It would take a lot of time and effort to extract valuable information from big datasets without visualization.

Simplifying complex data is one of the main reasons data visualization is crucial. Datasets of thousands or even millions of records are a common task for data scientists. It can be difficult to analyze such data in tabular form. Bar charts, line graphs, and histograms are examples of visualization approaches that summarize and display this data. Users can easily understand the general structure and important features of the data thanks to this simplification. For instance, a line chart can quickly display trends like growth or decline over time rather than examining a lengthy list of sales numbers.

The capacity of data visualization to identify patterns and trends is another important aspect. When examining raw data, patterns are frequently obscured. Relationships, seasonal trends, and recurrent patterns can all be found with the aid of visualization. A business might use a line graph, for example, to track monthly sales trends and pinpoint peak times. Scatter plots can also show relationships between variables, as the one between sales revenue and advertising expenditures. Finding these patterns is essential for forecasting future events and making well-informed judgments.

Finding anomalies and outliers is another important use of data visualization. Unusual data points that deviate greatly from other observations are known as outliers. These could point to significant uncommon occurrences or mistakes in data collecting. Box plots and scatter plots are examples of visualization methods that facilitate the identification of these anomalies. For instance, an unexpected increase or decrease in transactions in financial data could be a sign of fraud or system malfunctions. Early detection of these irregularities enables businesses to take corrective action and avert possible losses.

Effective communication requires not only analysis but also data display. Presenting their findings to stakeholders who might not have a technical expertise is a common task for data scientists. Communicating difficult concepts in an easy-to-understand manner is facilitated by visual representations. Decision-makers may immediately grasp important information without having to examine raw data thanks to charts and dashboards. A dashboard, for instance, can be used by a business management to track market trends, consumer behavior, and sales performance in real time.

The ability of data visualization to aid in decision-making is another crucial feature. To boost productivity, save expenses, and boost profitability, businesses rely on data-driven decision-making. Decision-makers may assess many possibilities and select the best course of action thanks to visualization, which offers a clear and succinct picture of facts. For example, a business can find popular products and modify its marketing efforts by using visualizations to analyze customer buying patterns.

Additionally, data visualization is essential for the exploratory data analysis (EDA) stage of data science. It's critical to comprehend the data before using statistical methods or machine learning models. By revealing distributions, missing values, and correlations between variables, visualization aids in the exploration of the dataset. This comprehension guarantees the selection of suitable models and methods for additional examination. Without adequate visualization, data scientists could miss crucial aspects of the data, producing unreliable findings.

Data visualization techniques and methods have become more sophisticated in contemporary data science. Users may generate dynamic and interactive visualizations with tools like Tableau, Power BI, and Python libraries. With the use of these tools, users may drill down into specifics, zoom, and filter data. Interactive graphics enable users to examine data from several angles and offer a better level of comprehension.

In data science, Python is one of the most popular programming languages. It offers robust visualization-making libraries. The simplest and most popular Python visualization library is called Matplotlib. It enables users to make basic graphs like scatter plots, line charts, bar charts, and histograms. Complete control over the labels, colors, and styles of graphs is provided by Matplotlib. Seaborn offers more sophisticated and visually appealing visualizations and is based on Matplotlib. For statistical analysis in particular, it is helpful. Seaborn makes it easier to create complicated graphs like distribution plots, pair plots, and heatmaps. Additionally, it offers pre-built practice datasets. Plotly library is used for creating interactive visualizations. Unlike static graphs, Plotly charts allow users to zoom, hover, and interact with data.

R is another popular language for statistical computing and visualization. ggplot2 is the most powerful visualization package in R. It follows the “Grammar of Graphics” concept, allowing users to build plots layer by layer.

Power BI is developed by Microsoft and is used for creating interactive dashboards and reports. It enables users to:

- in information from several sources
- Use Bring drag-and-drop to create visualizations
- Create dashboards for monitoring in real time.
- Another well-liked BI tool for data visualization is Tableau.

It offers:

- Interactive dashboards
- High-level analytics
- Simple integration of data

Check Your Progress-1

a. What do you mean by data visualization?

b. List out categories of Visualization with appropriate examples.

c. Explain comparison based visualization in details.

d. _____tool is developed by Microsoft and is used for creating interactive dashboards and reports.

e. How Visualization is imporstant for Exploratory Data Analysis?

f. Which libraries are used in Python for visualization?

4.3 Introduction to Data Interpretation

The process of examining, analysing, and explaining data in order to find patterns, extract valuable information, and aid in decision-making is known as data interpretation. To put it simply, data interpretation enables individuals to comprehend the true meaning of the obtained data. When properly analysed, raw data can become important knowledge that can be applied in business, science, education, healthcare, government, and many other disciplines, even though it may seem complex or meaningless on its own.

Large volumes of data are produced every second in the contemporary digital world by computers, mobile devices, websites, sensors, social media, banking systems, and online transactions. Businesses gather this information to better understand consumer behaviour,

enhance services, forecast future trends, and address issues. However, collecting data alone is not enough. The real value of data comes from interpreting it correctly. Therefore, data interpretation has become an essential part of data science, business analytics, and information technology.

Almost every sector and subject of research depends heavily on data interpretation. Data cannot yield valuable insights without appropriate interpretation.

Data Interpretation has numerous advantages:

- **Converts Raw Data into Meaningful Information**: Typically, raw data is disorganized and challenging to comprehend. Data interpretation provides a meaningful organization and explanation of the data. Statistical analysis, tables, graphs, and charts make complex material easier to comprehend.
- **Identifies Trends and Patterns**: Finding patterns, correlations, and trends in data is made easier with the aid of data interpretation. Organizations can forecast future results and comprehend variations throughout time with the aid of these patterns.
- **Supports Problem Solving :** Data interpretation aids in determining the root causes of issues and locating appropriate fixes. To identify problems and enhance performance, organizations examine operational data.
- **Improves Research and Analysis:** Data interpretation is crucial to research investigations in order to derive conclusions from gathered data. To validate hypotheses, uncover correlations between variables, and report findings, researchers employ interpretive techniques. Accurate data interpretation is critical to corporate surveys, social science research, and scientific investigations.

The following are the main goals of data interpretation:

- To arrange and condense gathered information
- To find significant trends and connections
- To transform data into comprehensible information
- To facilitate precise decision-making
- To forecast future results and patterns
- To address issues in research and business
- To enhance strategy formulation and planning

Data Interpretation play vital role in numerous applications.

Data interpretation is widely used by business companies to enhance operations, boost revenue, and comprehend market behavior. Businesses gather information on sales, consumer preferences, worker productivity, product demand, and market trends. Businesses can increase

overall productivity and make better strategic decisions by analyzing this data. Businesses can better comprehend market rivalry by using data interpretation. Businesses assess their strengths and shortcomings by comparing their performance to that of their rivals. This encourages more effective resource management, promotional efforts, and pricing methods. Data interpretation is utilized in contemporary e-commerce platforms to suggest products to consumers based on their past searches and purchases.

One of the most crucial industries with substantial uses for data interpretation is healthcare. Large volumes of patient-related data, including medical history, test results, treatment records, and illness statistics, are gathered by hospitals, clinics, and medical researchers. Better medical care is provided by physicians and other healthcare providers when this data is properly interpreted. In order to effectively identify illnesses and suggest appropriate therapies, doctors analyze patient data. To determine a patient's health status, for instance, blood test results, blood pressure readings, and temperature readings are analyzed. Monitoring the efficacy of medications and therapies is another benefit of data interpretation. In order to find new medications and vaccinations, medical researchers examine data from clinical trials. Healthcare institutions analyze data during disease outbreaks to forecast the spread of illnesses and implement preventative measures. For instance, in order to successfully manage public health systems during pandemics, governments and medical facilities examine infection rates, recovery rates, and vaccine statistics. By examining patient admittance records, bed availability, and healthcare resource consumption, data interpretation also aids hospital administration.

Data interpretation is used by educational institutions to enhance academic planning, student performance, and instructional strategies. Data about attendance, test scores, assignments, and student involvement are gathered by schools, colleges, and universities. Teachers and administrators can better understand student learning behavior and academic success by interpreting this data.Data interpretation is frequently used by educational institutions for academic research and curriculum improvement. Universities can enhance course designs and use cutting-edge teaching methods by examining student and instructor feedback. Interpreted data is used by online learning systems to monitor student participation and provide educational resources. Data interpretation facilitates the analysis of survey responses, experimental findings, and statistical data for scholarly investigations in higher education and research.

In order to manage financial operations and lower risks, banks and other financial organizations rely significantly on data interpretation. Financial institutions gather information about loans, investments, customer accounts, transactions, and market circumstances. Institutions can make

safe and profitable financial decisions by interpreting this data. Fraud detection is one significant application. To spot questionable activity like fraudulent transactions or strange account behavior, banks examine transaction trends. Banks are able to take preventive measures right once if anomalous trends are found. Banks use data interpretation to evaluate loan applications as well. Before granting loans, financial institutions examine the income, credit history, repayment history, and financial conduct of their clients. As a result, there is less chance of loan defaults.

Data interpretation is used by governments for public welfare initiatives, economic planning, and policy-making. Large volumes of information about the population, jobs, healthcare, education, agriculture, and economic activity are routinely gathered and examined. To comprehend employment statistics, literacy rates, and demographic shifts, population census data is analyzed. This interpretation allows governments to plan public services, transportation networks, healthcare facilities, and educational institutions. Governments can keep an eye on national income, taxes, unemployment, and inflation by using economic data analysis. This aids in the creation of economic policies and budget preparation. To comprehend voter behavior during elections, political analysts analyze popular sentiment and voting trends. In order to manage natural resources and deal with environmental issues like pollution, water scarcity, and climate change, governments also analyze environmental and climatic data.

One of the fundamental tasks in data science and artificial intelligence (AI) is data interpretation. Through digital platforms, sensors, websites, and social media, modern enterprises produce enormous volumes of both organized and unstructured data. In order to produce insights and create intelligent systems, data scientists analyze this data. In order to find patterns and provide predictions, machine learning algorithms rely on interpreted data. For instance, streaming services use user viewing activity to suggest films and television series. In a similar vein, internet retailers use consumer behavior to make product recommendations.

4.4 Types and Methods of Data Interpretation

Data interpretation can generally be divided into two major categories:

- **Qualitative Data Interpretation:** The process of examining, understanding, and explaining non-numerical data to find meanings, patterns, opinions, experiences, and connections is known as qualitative data interpretation. Qualitative data interpretation deals with descriptive information including words, perceptions, feelings, behaviors, and experiences, in contrast to quantitative data interpretation, which concentrates on numbers

and statistical computations. Understanding the deeper meaning underlying human ideas, behaviors, and interactions is the main goal of qualitative interpretation. In many real-world scenarios, a problem or phenomena cannot be fully explained by numerical data alone. Exam results, for instance, may indicate that students are performing poorly in a topic, but qualitative interpretation can assist pinpoint the causes of low performance, such as stress, lack of interest, or challenges with instruction. In order to comprehend complicated social, psychological, educational, and organizational challenges, qualitative interpretation is crucial. Many disciplines, including the social sciences, education, healthcare, psychology, marketing, business management, and research projects, make extensive use of qualitative data. It enables academics and companies to gain a deeper understanding of human behavior, consumer opinions, emotional reactions, and social encounters. Non-numerical information that characterizes traits, attributes, experiences, or observations is referred to as qualitative data. This kind of data can be observed, documented, and interpreted using words and descriptions, but it is typically not quantifiable using numbers. Non-numerical information that characterizes traits, attributes, experiences, or observations is referred to as qualitative data. This kind of data can be observed, documented, and interpreted using words and descriptions, but it is typically not quantifiable using numbers. Interpreting qualitative data differs from interpreting quantitative data in a number of significant ways. It has descriptive nature as it emphasizes utilizing words rather than numbers to convey experiences, beliefs, feelings, and behaviours. Rather than computing averages or percentages, the interpretation process entails comprehending meanings and situations. Qualitative interpretation is frequently subjective since it may rely on the researcher's or analyst's knowledge and viewpoint. The same data may be interpreted differently by various people depending on their perspectives and experiences. The other characteristic is it focus on human experiences and emotions. Qualitative interpretation studies data within its context. Instead of only identifying facts, it attempts to understand the surrounding conditions and reasons behind behaviours or events. It is also flexible in nature as researchers can modify questions and approaches during the study.

- **Quantitative Data Interpretation:** The process of examining, analyzing, and explaining numerical data in order to draw significant conclusions and aid in decision-making is known as quantitative data interpretation. It emphasizes on quantifiable data that can be expressed using tables, graphs, computations, statistical techniques, and numbers. Through mathematical and statistical analysis, quantitative interpretation aids researchers, corporations, governments, and organizations in comprehending trends, correlations,

patterns, and variances in data. Huge volumes of numerical data are produced daily by corporate transactions, scientific research, healthcare systems, educational institutions, financial operations, social media platforms, and internet apps in today's digital and technologically advanced world. Only when this data is correctly interpreted does it become valuable. In order to address issues, enhance performance, and forecast future results, quantitative data interpretation converts unprocessed numerical values into comprehensible insights. Quantitative data refers to numerical information that can be measured, counted, or expressed mathematically. This type of data deals with quantities, amounts, sizes, frequencies, percentages, and measurements. The fact that quantitative interpretation works with numerical quantities is its most significant feature. Numbers are used to represent data, which facilitates calculation, comparison, and analysis. Statistical techniques including averages, percentages, correlation, regression, and probability analysis are used in quantitative interpretation to scientifically analyze data. Quantitative data is usually structured and organized into tables, spreadsheets, databases, and charts. This makes analysis systematic and efficient. Quantitative interpretation is highly useful for analyzing large volumes of data because numerical information can be processed quickly using computers and statistical software. Quantitative data is generally divided into two major categories. Discrete data consists of countable numerical values that usually cannot be divided into fractions or decimals. Continuous data consists of measurable values that can take any numerical value within a range, including decimals and fractions.

Several methods are used to interpret quantitative data effectively. Tabular interpretation is most vital method for data interpretation. One of the best ways to systematically arrange and show data is through tabular format. Tables facilitate proper interpretation, make analysis easier, and enhance readability by organizing data into rows and columns. They are extensively utilized for information storage, comparison, and analysis in business, government, education, healthcare, and data science. A well-designed table minimizes confusion and complexity while assisting users in rapidly identifying trends, linkages, and patterns. Tables are still a crucial tool in statistics, data analysis, and contemporary computer systems despite possible drawbacks. Understanding tabular format is crucial for undergraduate students because it fosters the organizing, analytical, and data-handling skills needed for both professional and academic employment.In modern computing and data science, tabular data is extremely important. Databases, spreadsheets, and programming tools use tabular structures extensively.

Following are the best examples of tabular form:

- Microsoft Excel spreadsheets
- SQL database tables
- Python Pandas DataFrames
- R data frames

Data scientists use tabular datasets for:

- Data cleaning
- Statistical analysis
- Machine learning
- Visualization

Tabular data forms the foundation of many analytical and computational processes.

Tables help summarize:

- Sales figures
- Student marks
- Financial records

Graphical Interpretation is another important method for Data Interpretation.The process of comprehending, evaluating, and elucidating data using visual aids like graphs, charts, diagrams, and plots is known as graphic interpretation. Graphical interpretation transforms information into visual formats that make complex data easier to comprehend and evaluate, rather than merely providing it in numerical or verbal form. It facilitates the fast and efficient identification of patterns, correlations, trends, comparisons, and variances. Graphical interpretation is crucial to modern data analysis and data science because people can comprehend visual information more readily than lengthy paragraphs or big tables. Graphical methods are used by governments, businesses, scientists, academics, and educators to support decision-making processes and effectively convey data. Graphical interpretation refers to the analysis and explanation of data presented in graphical form. It involves observing graphs or charts carefully to identify important features such as:

- Trends
- Comparisons
- Growth or decline
- Relationships between variables
- Distribution patterns
- Maximum and minimum values
- Simple integration of data

Check Your Progress-2

a. What do you mean by data interpretation?

b. What are the advantages of data interpretation?

c. Explain role of data interpretation in education sector.

d. What do you mean by qualitative data interpretation?

e. Explain data interpretation methods for quantitative data.

4.5 Let us sum up

In this unit we have discussed overview of data interpretation in details. The unit also discussed the advantages of it in details. The unit also discussed the role of data interpretation in various applications. The unit also discussed various methods for data interpretation in details.

4.6 Check your progress: Possible Answers

1-a The process of displaying data in a graphical or visual manner, such as charts, graphs, and maps, is known as data visualization.

1-b

Categories of Visualization	**Example**
Comparison	Bar,Column
Trend	Line,Area
Distribution	Histogram, Boxplot
Relationship	Scatter,Bubble
Composition	Pie, Staked Bar
Hierarchy	Tree Maaps
Geographical	Map

1-c One of the most popular ways to depict data is through comparison-based visualizations. By comparing values across various categories, these visualizations make it simple to spot variances and similarities. The bar chart, which uses rectangular bars to depict data values, is a common example of this genre. Each bar's height or length reveals the size of the value it represents. For example, a bar chart offers a quick and easy visual comparison when a teacher needs to compare students' grades in several disciplines. Comparing monthly or annual performance is another common usage for column charts, which are just vertical bar charts, in business reporting.

1-d Power BI

1-e Data visualization is essential for the exploratory data analysis (EDA) stage of data science. It's critical to comprehend the data before using statistical methods or machine learning models. By revealing distributions, missing values, and correlations between variables, visualization aids in the exploration of the dataset. This comprehension guarantees the selection of suitable models and methods for additional examination. Without adequate visualization, data scientists could miss crucial aspects of the data, producing unreliable findings.

2-a The process of examining, analysing, and explaining data in order to find patterns, extract valuable information, and aid in decision-making is known as data interpretation

2-b Data Interpretation has numerous advantages:

Converts Raw Data into Meaningful Information: Typically, raw data is disorganized and challenging to comprehend. Data interpretation provides a meaningful organization and explanation of the data. Statistical analysis, tables, graphs, and charts make complex material easier to comprehend.

Identifies Trends and Patterns: Finding patterns, correlations, and trends in data is made easier with the aid of data interpretation. Organizations can forecast future results and comprehend variations throughout time with the aid of these patterns.

Supports Problem Solving : Data interpretation aids in determining the root causes of issues and locating appropriate fixes. To identify problems and enhance performance, organizations examine operational data.

Improves Research and Analysis: Data interpretation is crucial to research investigations in order to derive conclusions from gathered data. To validate hypotheses, uncover correlations between variables, and report findings, researchers employ interpretive techniques. Accurate data interpretation is critical to corporate surveys, social science research, and scientific investigations.

2-c Data interpretation is used by educational institutions to enhance academic planning, student performance, and instructional strategies. Data about attendance, test scores, assignments, and student involvement are gathered by schools, colleges, and universities. Teachers and administrators can better understand student learning behavior and academic success by interpreting this data.Data interpretation is frequently used by educational institutions for academic research and curriculum improvement. Universities can enhance course designs and use cutting-edge teaching methods by examining student and instructor

feedback. Interpreted data is used by online learning systems to monitor student participation and provide educational resources. Data interpretation facilitates the analysis of survey responses, experimental findings, and statistical data for scholarly investigations in higher education and research.

2-d The process of examining, understanding, and explaining non-numerical data to find meanings, patterns, opinions, experiences, and connections is known as qualitative data interpretation.

2-e Several methods are used to interpret quantitative data effectively. Tabular interpretation is most vital method for data interpretation. One of the best ways to systematically arrange and show data is through tabular format. Tables facilitate proper interpretation, make analysis easier, and enhance readability by organizing data into rows and columns. They are extensively utilized for information storage, comparison, and analysis in business, government, education, healthcare, and data science. A well-designed table minimizes confusion and complexity while assisting users in rapidly identifying trends, linkages, and patterns. Tables are still a crucial tool in statistics, data analysis, and contemporary computer systems despite possible drawbacks. Understanding tabular format is crucial for undergraduate students because it fosters the organizing, analytical, and data-handling skills needed for both professional and academic employment.In modern computing and data science, tabular data is extremely important. Databases, spreadsheets, and programming tools use tabular structures extensively.

Following are the best examples of tabular form:

Microsoft Excel spreadsheets

SQL database tables

Python Pandas DataFrames

R data frames

Data scientists use tabular datasets for:

Data cleaning

Statistical analysis

Machine learning

Visualization

Tabular data forms the foundation of many analytical and computational processes.

Tables help summarize:

Sales figures

Student marks Financial records

4.7 Further Reading

1. https://www.geeksforgeeks.org/data-visualization/data-visualization-and-its-importance/
2. https://www.thoughtspot.com/data-trends/data-visualization
3. https://www.mygreatlearning.com/blog/what-is-data-interpretation/

4.8 Assignments

1. Why Data Visualization is important in real life?
2. What is the role of trend visualization?
3. Explain advantages of Data Visualization
4. How Data Visualization is effective in Decision Making?
5. What are the advantages of Data Interpretation?
6. Explain Quantitative Data Analysis in details.
7. How Quantitative Data Interpretation is different than Qualitative Data Interpretation?
8. Explain Graphical Data Interpretation method in details.

Block-4

Introduction to Data Science Tools

Unit-1: Programming Languages for Data Science

Unit Structure

1.0 Learning Objectives

After studying this unit, the student should be able to:

- Understand the role of a programming language in the data science process.
- Explain why Python is the most popular language for data science.
- Describe how Python is used at each step of the data science workflow.
- List and explain the main advantages of Python.
- Understand the role of R in data science, especially in statistical analysis.
- Compare Python and R and decide which language is suitable for which task.
- Get a basic idea of other languages like SQL, Julia, Java and Scala that are also used in data science.

1.1 Introduction

Data science is the area of study that helps us turn raw data into useful information. For example, when an e-commerce website like Flipkart or Amazon recommends a product to you, when YouTube suggests the next video, or when your bank flags a suspicious transaction, all of this is done using data science. Behind every such system, there is a person, called a data scientist, who has collected data, cleaned it, analysed it and built a model that gives the result. To do all this work, the data scientist needs a tool. That tool is a programming language. A programming language is simply a set of rules and instructions that lets us tell the computer what to do. Just as we use English, Hindi or Gujarati to talk to other people, we use a programming language to talk to a computer.

In data science, a programming language is used at every stage. We use it to read the data from a file, to clean the data, to perform calculations, to draw charts and to build prediction models. Without a programming language, none of this would be possible in a fast and reliable way.

There are many programming languages available, but two of them have become the most popular for data science. They are Python and R. Almost every data science book, online course and job description today mentions one or both of these. In this unit, we will learn what these languages are, why they are popular and how they help in data science. We will also briefly look at a few other languages that are useful in this field.

1.2 Why do we need a Programming Language in Data Science?

Before we look at any specific language, let us first understand why we need a programming language in the first place. Some students often ask, "Why can’t we just use Microsoft Excel

for everything?" Excel is a great tool, and it is fine for small datasets, say a few thousand rows. But real-world datasets often contain millions of rows, come from many different sources and need complex calculations. Excel becomes slow or simply cannot open such files. A programming language solves this problem. The main roles of a programming language in data science are listed below.

1. Data Collection: Data can come from many places, such as a file on your computer, a database, a website or even a mobile application. A programming language gives us a way to collect data from all these sources easily and bring it into one place.

2. Data Cleaning and Pre-processing: Real-world data is usually messy. Some values may be missing, some may be entered incorrectly, and some rows may be duplicates. Before we can analyse the data, we have to clean it. Programming languages provide tools to find errors, fill in missing values and remove duplicates very quickly.

3. Exploratory Data Analysis (EDA): Once the data is clean, we want to look at it carefully to understand what it contains. We may want to find the average marks in a class, the highest sale of the month, or the most popular item on a website. EDA is the process of exploring the data using simple statistics and charts. Programming languages make this very easy.

4. Statistical Analysis: Statistics tells us about patterns in the data. For example, we may want to find out whether students who attend more classes get better marks. Such questions are answered using statistical techniques like correlation, regression and hypothesis testing. Programming languages have ready-made functions for all these techniques.

5. Building Machine Learning Models: Machine learning is the part of data science where we teach the computer to learn from data and make predictions. For example, the computer can learn from past house sale data and then predict the price of a new house. Programming languages provide ready-made libraries to build such models without writing the entire algorithm from scratch.

6. Data Visualization: A picture is worth a thousand words. The same is true for data. A good chart can show a pattern that a long table cannot. Programming languages let us create different types of charts such as bar charts, line charts, pie charts and even interactive dashboards.

7. Working with Big Data: When the data becomes very large, normal techniques become slow. There are special programming tools that can split the data into many parts and process all parts at the same time, on many computers. This is called distributed computing.

8. Automation and Deployment: After we build a model, we usually want to use it again and again. We may want it to run every night automatically. Programming languages let us automate such tasks and deploy our models on the internet so that other people can use them.

9. Cloud Integration: Many companies today store their data on cloud platforms such as Amazon Web Services (AWS), Google Cloud Platform (GCP) and Microsoft Azure. Programming languages provide easy ways to connect to these cloud services.

From the above points, we can see that a programming language is used throughout the data science life cycle. The two most widely used programming languages in data science are Python and R. We will now study both of them in detail.

1.3 Python Programming Language

Python is a high-level, general-purpose programming language. It was created by Guido van Rossum and first released in 1991. The name Python is taken from a comedy show called "Monty Python's Flying Circus" and not from the snake. Today, Python is used everywhere, from website development and game programming to artificial intelligence and data science.

In data science, Python has become the number one choice. There are three main reasons for this. First, Python's syntax is very simple and looks almost like plain English. Even a beginner can read a Python program and understand what it does. Second, Python has thousands of free libraries that are made specifically for data science work. A library is a collection of pre-written code that we can use directly, without writing it ourselves. Third, Python has a very large and active community of users who share their code and answer questions on websites like Stack Overflow.

To give you a feel of how simple Python is, look at the following small program that prints a message:

```
# A simple Python program
name = 'BCA Student'
print('Hello,', name)
```

When this code is run, the output is: Hello, BCA Student. As you can see, the code is short and easy to understand. There is no need to declare the type of the variable name, no semicolons at the end of lines, and no curly braces. This simplicity is one of Python's greatest strengths

1.3.1 Role of Python in Data Science

Python is used in every stage of the data science workflow. Let us look at each stage and see how Python helps.

- Data Collection: Python provides several tools for collecting data from different sources. For example, the requests library lets us download data from a website, and the BeautifulSoup library helps us extract information from web pages, a process called web scraping. Python can also connect to databases like MySQL, Oracle and MongoDB to fetch

data directly. The result is that we can pull data from almost any source into our program with just a few lines of code.

- Data Cleaning and Pre-processing: Once data is collected, it is rarely ready for analysis. There may be missing values, wrong entries or different formats for the same thing (for example, "Mumbai", "mumbai" and "MUMBAI"). Python's Pandas and NumPy libraries are very powerful for cleaning such data. They allow us to remove rows with missing values, fill empty cells with default values, change text to lower case and convert one data type to another. Scikit-learn provides further pre-processing options such as scaling numbers to a common range and encoding text categories into numbers.
- Exploratory Data Analysis (EDA): EDA is the process of getting to know your data. With Python, you can quickly find the average, minimum, maximum, count and other summary statistics of any column. You can also draw charts to see the distribution of values. Libraries like Pandas, Matplotlib and Seaborn are commonly used for EDA. For example, with one line of Python code you can find out how many students passed and how many failed in an exam.
- Building and Testing Models: Python's Scikit-learn library has ready-made implementations of all the standard machine learning algorithms, such as linear regression, decision trees, k-nearest neighbours and clustering. For more advanced work involving deep learning, libraries like TensorFlow, Keras and PyTorch are used. With these libraries, we can train a model on a dataset and test how well it performs, all in a few lines of code.
- Model Evaluation and Tuning: After building a model, we need to check how good it is. Python provides functions to compute accuracy, precision, recall and many other measures. It also lets us improve the model by trying different parameters automatically, a process known as hyperparameter tuning.
- Deployment: Once a model is ready, we may want to make it available to users through a website or a mobile app. Python frameworks like Flask and Django can be used to build such websites. Newer tools like Streamlit and Dash let us quickly build interactive dashboards where the user can change the inputs and see the results updated in real time.

1.3.2 Important Python Libraries for Data Science

As mentioned earlier, the real power of Python comes from its libraries. Some of the most important libraries for data science are listed below. We will study most of these in detail in the next unit.

- NumPy: For numerical computing and working with arrays.

- Pandas: For working with structured data in the form of tables.
- Matplotlib: For drawing different types of charts and graphs.
- Seaborn: For drawing attractive statistical charts.
- Scikit-learn: For machine learning algorithms.
- TensorFlow and Keras: For deep learning.
- BeautifulSoup and Scrapy: For web scraping.
- Statsmodels: For advanced statistical analysis.

1.3.3 Advantages of Python

1. Easy to Learn and Read: Python's syntax is very close to English. A beginner who has never seen Python before can often guess what a Python program does just by reading it. This means students can spend more time learning data science concepts and less time fighting with the language.

2. Free and Open Source: Python is completely free to download and use, even for commercial purposes. The source code is also open, which means anyone can see how it is built and contribute improvements.

3. Huge Library Support: Python has more than 200,000 free libraries available on the Python Package Index (PyPI). This means that for almost any task you can think of, there is already a library that does it. You do not have to write everything from scratch.

4. Strong Community: Python has one of the largest programming communities in the world. If you face a problem, the chances are very high that someone else has faced the same problem before and has shared the solution online. Websites like Stack Overflow and GitHub have millions of Python questions and example programs.

5. Platform Independent: Python programs run on Windows, macOS and Linux without any change. You can write a program on your laptop and run it on a server in the cloud without making any modifications.

6. Suitable for All Stages: From data collection to deployment, Python supports every stage of the data science process. This means we do not have to switch between many languages during a project.

7. Integration with Other Tools: Python can connect easily with databases, web services, big data systems and cloud platforms. It can also call code written in C, C++ and Java when very high speed is needed.

8. Supports Multiple Programming Styles: Python supports object-oriented programming, procedural programming and functional programming. This makes it flexible and suitable for different types of problems.

Check Your Progress-1

List any four roles of programming languages in the data science process.

Why is Python considered the most popular language for data science? Give at least three reasons.

Which Python library is used for: (a) data manipulation, (b) numerical computing, (c) machine learning, (d) plotting charts?

Is Python platform independent? Justify your answer.

Name any two Python frameworks that can be used to deploy a machine learning model.

1.4 R Programming Language

R is another very popular language in data science. It was created by two statisticians, Ross Ihaka and Robert Gentleman, at the University of Auckland in New Zealand in the early 1990s. The name R comes from the first letters of their first names. R was designed from the very beginning for statistical computing and graphics. Because of this, R has very strong support for statistical analysis and data visualization.

While Python is a general-purpose language that became popular in data science, R is a special-purpose language that was built for data analysis. This is the main reason why R is still very widely used in research, especially in the fields of statistics, biology, medicine, social sciences, economics and finance. Many universities use R to teach statistics.

R is free and open source, just like Python. It is available for Windows, macOS and Linux. The standard way to write R programs is to use a tool called RStudio, which is a free integrated development environment (IDE) made specifically for R.

Here is a simple R program to give you a feel of the language:

```
# A simple R program
marks <- c(85, 90, 78, 92, 88)
average <- mean(marks)
print(average)
```

This program creates a list of marks, calculates the average and prints it. As you can see, R also has a very clean syntax. The c() function is used to combine values into a vector, and mean()

is a built-in function to calculate the average. R has hundreds of such built-in functions for statistics.

1.4.1 Role of R in Data Science

R, like Python, is used at many stages of data analysis. However, the strength of R is mainly in statistics and visualization. Let us see how R helps at different stages.

Data Collection and Cleaning: R has many packages for cleaning data. The dplyr package provides simple commands to filter rows, select columns, change values and create new variables. The stringr package helps in working with text data. The lubridate package makes it very easy to work with dates and times. Together, these packages give R very powerful data wrangling capabilities, which means the ability to reshape and clean data easily.

Exploratory Data Analysis (EDA): R has excellent tools for EDA. The base R package itself can produce histograms, scatter plots and box plots with very little code. The famous ggplot2 package can produce publication-quality charts that are used in research papers and books. With ggplot2, we can build a chart layer by layer, adding points, lines, colours and labels as we go.

Statistical Analysis: This is where R truly shines. R was built by statisticians for statisticians. It supports almost every statistical method ever invented, from simple t-tests and chi-square tests to advanced techniques like Bayesian analysis, time series forecasting and survival analysis. Packages like MCMCpack, brms and survival are widely used in research.

Model Training and Evaluation: The caret package in R provides a unified interface to train and evaluate machine learning models. It supports cross-validation, parameter tuning and many performance measures. Packages like ROCR and pROC are used to evaluate classification models using ROC curves.

Reporting and Deployment: R has some of the best reporting tools in the data science world. R Markdown lets us mix code, output and explanation in a single document, which can then be exported as a PDF, Word file or web page. Shiny is a package that lets us build interactive web applications without any web development knowledge. Plumber lets us turn R functions into web APIs.

1.4.2 Important R Packages

R has a huge collection of packages available on CRAN (Comprehensive R Archive Network), which is the official repository of R packages. As of 2024, CRAN had more than 20,000 packages. Some of the most popular ones for data science are:

- dplyr: For data manipulation.
- tidyr: For reshaping and tidying data.
- ggplot2: For creating beautiful charts and graphs.
- stringr: For text manipulation.
- lubridate: For working with dates and times.
- caret: For machine learning.
- shiny: For building interactive web applications.
- rmarkdown: For creating reports and documents.

1.4.3 Advantages of R in Data Science

1. Specialized for Statistics: R was created by statisticians, for statisticians. Whatever statistical method you can think of, there is most probably an R package for it. This makes R the best choice when the work involves heavy statistical analysis.

2. Excellent Visualization: R's ggplot2 library is considered one of the best data visualization tools in the world. The charts made with ggplot2 are used in research journals, government reports and even in newspapers.

3. Reproducible Research: R Markdown allows us to create documents that contain code, results and explanation together. If we run the document again later, we get exactly the same results. This is very useful in research, where reproducibility is very important.

4. Strong Academic and Research Community: Most new statistical methods are first implemented in R, often by the very researchers who invented them. So if you want to use the latest statistical techniques, R is usually the first language to support them.

5. Free and Open Source: R is completely free. There are no license fees, even for use in companies.

6. Active Community: There is a large and helpful community of R users worldwide. Problems and solutions are shared on forums like Stack Overflow and the RStudio Community.

7. Easy Reporting and Sharing: With R Markdown and Shiny, sharing results with others (even non-technical people) becomes very simple. We can build dashboards and reports that update automatically when the data changes.

1.4.5 Advantages of R in Data Science

1. Specialized for Statistics: R was created by statisticians, for statisticians. Whatever statistical method you can think of, there is most probably an R package for it. This makes R the best choice when the work involves heavy statistical analysis.

2. Excellent Visualization: R's ggplot2 library is considered one of the best data visualization tools in the world. The charts made with ggplot2 are used in research journals, government reports and even in newspapers.

3. Reproducible Research: R Markdown allows us to create documents that contain code, results and explanation together. If we run the document again later, we get exactly the same results. This is very useful in research, where reproducibility is very important.

4. Strong Academic and Research Community: Most new statistical methods are first implemented in R, often by the very researchers who invented them. So if you want to use the latest statistical techniques, R is usually the first language to support them.

5. Free and Open Source: R is completely free. There are no license fees, even for use in companies.

6. Active Community: There is a large and helpful community of R users worldwide. Problems and solutions are shared on forums like Stack Overflow and the RStudio Community.

7. Easy Reporting and Sharing: With R Markdown and Shiny, sharing results with others (even non-technical people) becomes very simple. We can build dashboards and reports that update automatically when the data changes.

Check Your Progress-2

Who created the R programming language and where?

What is CRAN and why is it important for R users?

Name any three R packages and state their purpose.

Which R package is most popular for data visualization?

What is R Markdown and why is it useful?

1.5 Python vs R: A Simple Comparison

Both Python and R are excellent languages for data science. Many beginners get confused about which one to learn first. The truth is, both languages can do most data science tasks. However, there are some differences that make each language better suited for certain situations. The table below compares the two languages on several points.

Point of Comparison	Python	R
Year created	1991	1993
Type of language	General-purpose	Made for statistics

Point of Comparison	Python	R
Ease of learning	Very easy, English-like syntax	Easy for those with statistics background
Main strength	Machine learning, deployment, web applications	Statistical analysis, research, visualization
Library/package source	PyPI (Python Package Index)	CRAN (Comprehensive R Archive Network)
Popular libraries	NumPy, Pandas, Scikit-learn, TensorFlow	dplyr, ggplot2, caret, shiny
Visualization	Matplotlib, Seaborn, Plotly	ggplot2 (very strong)
Best for	End-to-end projects from data collection to deployment	Detailed statistical research and reports
Industry use	Very widely used in companies	Common in research, academics, healthcare and finance
Used in cloud and big data	Yes, very well supported	Possible but less common
Job market	Very large number of jobs	Many jobs, mostly in research and analytics roles

From the above table, we can give a simple recommendation. If you are a BCA student who is just starting with data science and you also want to learn skills useful for software jobs, start with Python. Python will help you in machine learning, web development and many other areas. If you are more interested in research, statistics or fields like biology and economics, R is also a very good choice. In practice, many data scientists know both languages and use whichever is better for the task at hand.

1.6 Other Useful Languages: SQL, Julia, Java and Scala

Apart from Python and R, a few other languages are also commonly used in data science. As a BCA student, you should be aware of them, even though we will not study them in detail in this block.

1.6.1 SQL (Structured Query Language)

SQL is a special language used to talk to databases. Almost all the data in the world is stored in some kind of database. Whether it is the data of a bank, a hospital, a college or an e-commerce website, the data is most probably in a database like MySQL, Oracle, PostgreSQL or SQL Server. Here is a small example of an SQL query that fetches the names of all students whose marks are greater than 80:

```
SELECT name FROM students WHERE marks > 80;
```

1.6.2 Julia

Julia is a relatively new programming language, first released in 2012. It was designed to be as easy to write as Python but as fast as C. Julia is becoming popular in scientific computing, simulations and high-performance numerical work.

1.6.3 Java

Java is one of the oldest and most widely used programming languages in the world. While Java is not commonly used for data analysis, it is very important in big data systems. Famous big data tools like Apache Hadoop and Apache Spark are built on Java. So if you ever work with big data systems, knowing Java is helpful.

1.6.4 Scala

Scala is a language that runs on the Java Virtual Machine (JVM) and is the main language used in Apache Spark, the most popular big data processing tool. If you want to work as a big data engineer, learning Scala along with Python is a good idea.

1.6.5 SAS

SAS (Statistical Analysis System) is a commercial software used in many large companies, especially in banking, insurance and pharmaceuticals. Unlike Python and R, SAS is not free. However, it is still used in many traditional companies because of its long history and reliable support.

1.7 Let's Sum Up

In this unit, we have learned that data science is a process that turns raw data into useful information, and that programming languages are the main tools used in this process. We then studied Python in detail and learned that it is the most popular language for data science because of its simple syntax, large library support, strong community and platform independence. Next, we studied R, which is a special-purpose language designed for statistics. We saw that R has very strong support for statistical analysis and data visualization, and is widely used in research

and academia. We also briefly looked at other languages such as SQL, Julia, Java, Scala and SAS, and explained where each is used. By the end of this unit, you should have a clear idea of which language to use for which kind of data science task.

1.8 Check your progress: Possible Answers

Roles of programming languages in data science:
(a) Data collection from various sources, (b) Data cleaning and pre-processing to handle missing values and errors, (c) Exploratory data analysis to understand patterns,
(d) Statistical analysis to test hypotheses, (e) Building machine learning models,
(f) Data visualization, (g) Working with big data, (h) Automation and deployment.

Python is the most popular language for data science because (a) its syntax is simple and easy to learn, (b) it has thousands of free libraries made for data science, (c) it has a large and active community of developers, (d) it is platform-independent, and (e) it supports the entire data science workflow from data collection to deployment.
Python libraries: (a) Pandas is used for data manipulation, (b) NumPy is used for numerical computing, (c) Scikit-learn is used for machine learning, (d) Matplotlib is used for plotting charts.

Yes, Python is platform-independent. The same Python program can run on Windows, macOS and Linux without any change. This is because Python is an interpreted language and Python interpreters are available on all major operating systems.

Two Python frameworks for deploying machine learning models are Flask and Django. In addition, Streamlit and Dash are popular for building interactive dashboards quickly.
R was created by Ross Ihaka and Robert Gentleman at the University of Auckland in New Zealand in the early 1990s.

CRAN stands for Comprehensive R Archive Network. It is the official repository of R packages. R users can download and install thousands of free packages from CRAN to extend the functionality of R for almost any kind of analysis.

Three R packages: (a) dplyr - for data manipulation, (b) ggplot2 - for data visualization, (c) caret - for machine learning. (Other valid answers include stringr, lubridate, shiny, rmarkdown.)

ggplot2 is the most popular R package for data visualization. It is used to create high-quality charts that are widely used in research and reports.

R Markdown is a tool that lets us mix R code, output and explanatory text in a single document. It is useful because it allows us to create reproducible reports that can be exported as PDF, Word, HTML or even slide presentations. The same document can be re-run any time to update the results automatically.

1.9 Further Reading

1. Wes McKinney, "Python for Data Analysis", O'Reilly Media, 3rd Edition, 2022.
2. Hadley Wickham and Garrett Grolemund, "R for Data Science", O'Reilly Media, 2nd Edition, 2023.
3. Reema Thareja, "Data Science and Machine Learning using Python", McGraw Hill, 2022.
4. Dharmendra Patel, Sanskruti Patel and Atul Patel, "Fundamentals of R Programming: A Practical Way", Sting Production, 2024.
5. Online resource: Official Python documentation at https://docs.python.org/3/
6. Online resource: Official R documentation at https://www.r-project.org/.

1.10 Assignments

1. Explain the role of programming languages in the data science process. Why can we not use only Microsoft Excel for all data science work?
2. List and explain the main advantages of Python that make it the most popular language in data science.
3. Describe how Python is used at each stage of the data science workflow. Give at least one Python library used at each stage.
4. What is R? Explain the role of R in data science and list any five important R packages with their use.

5. Compare Python and R on at least six different points. Based on your comparison, suggest which language a beginner should learn first and justify your answer.
6. Briefly explain the role of SQL, Julia, Java and Scala in data science. Why SQL is considered an essential skill for every data scientist?

Unit-2: Data Science Libraries and Frameworks

Unit Structure

2.0 Learning Objectives

After studying this unit, the student should be able to:

- Understand what a library is in Python and why libraries are useful.
- Explain the features and uses of the NumPy library.
- Use the Pandas library for working with structured data.
- Create simple charts and graphs using Matplotlib.
- Use Seaborn to create attractive statistical charts.
- Understand the role of Scikit-learn in machine learning.
- Know about other useful libraries such as SciPy, Statsmodels, TensorFlow and Keras.
- Understand how all these libraries work together in a data science project.

2.1 Introduction

In the previous unit, we learned that Python is the most popular language for data science. We also said that the real power of Python lies in its libraries. In this unit, we will study these libraries in detail. By the end of this unit, you will know which library to use for which task. Imagine you are building a house. You can certainly try to make every brick, every nail and every wire by yourself. But that would take a very long time. It is much easier to buy ready-made bricks, nails and wires from a shop. In the same way, when we write a program, we do not have to write every small piece of code from scratch. We can use ready-made code written by others. This ready-made code is called a library.

In Python, a library is simply a collection of pre-written functions and classes that we can use in our own programs. To use a library, we first install it (usually using a tool called pip) and then we import it into our program with the import statement. Once imported, we can use any function from the library by simply writing its name.

There are many libraries in Python that are useful for data science. The five most important ones are NumPy, Pandas, Matplotlib, Seaborn and Scikit-learn. Together, these five libraries cover most of what a data scientist needs to do. We will study each of them in this unit.

2.2 What is a Library and Why Do We Use It?

Before we look at specific libraries, let us first answer a few common questions about libraries in general.

2.2.1 Definition

A library is a collection of code (functions, classes and modules) that is written and maintained by someone else and made available for everyone to use. We do not have to know how the code inside the library works. We just need to know what each function does and how to call it.

2.2.2 Why Use Libraries?

Using libraries gives us many benefits.

1. Saves Time: Without libraries, we would have to write a lot of code from scratch. With libraries, we can write the same program in a fraction of the time.

2. Tested and Reliable: Popular libraries are used by millions of people. Bugs are quickly found and fixed. So library code is usually more reliable than the code we write ourselves.

3. Optimized for Speed: Many libraries are written in C or C++ underneath, even though we use them from Python. This makes them very fast. NumPy, for example, is much faster than plain Python lists.

4. Standard Way of Doing Things: When everyone uses the same library, code becomes easier to share and understand. A Python data scientist anywhere in the world will recognize Pandas code immediately.

5. Wide Range of Features: A single library may contain hundreds or thousands of functions covering different needs.

2.2.3 How to Install and Use a Library

To install a library in Python, we use a tool called pip, which stands for "Pip Installs Packages". For example, to install Pandas, we open the command prompt or terminal and type:

```
pip install pandas
```

To use a library in our program, we use the import statement at the top of our code. For example:

```
import pandas as pd
import numpy as np
```

In the above lines, we have imported pandas with the short name pd and numpy with the short name np. These short names are commonly used by data scientists, so it is a good idea to follow this convention. Once imported, we can use any function from the library by writing the short name followed by a dot and the function name. For example, pd.read_csv() reads a CSV file, and np.mean() calculates the mean.

2.2.4 General Features of Python Libraries

Python libraries used in data science share some common features that make them easy to use.

- Modular Design: We can import only the parts of the library that we need, which saves memory.
- Wide Coverage: There are libraries for data manipulation, numerical computing, visualization, machine learning, deep learning and statistical analysis.
- Compatibility with Each Other: Most libraries work well together. For example, we can pass a NumPy array to a Pandas function, and a Pandas DataFrame to a Matplotlib function.
- High-Level APIs: The functions are easy to call without knowing the underlying algorithms.
- Strong Data Structures: Libraries provide special data structures like NumPy ndarrays and Pandas DataFrames that handle data more efficiently than plain Python lists.
- Good Documentation: Most libraries have detailed websites with examples and tutorials.
- Active Community: Bugs are reported, fixes are released, and new features are added regularly.
- Open Source: Most libraries are free, and the source code is publicly available.

2.3 NumPy (Numerical Python)

NumPy stands for Numerical Python. It is the most fundamental library for numerical computing in Python. Almost every other data science library in Python is built on top of NumPy. So even if you do not use NumPy directly very often, you will use it indirectly all the time.

The main reason for NumPy's popularity is its powerful data structure called the ndarray, which stands for n-dimensional array. An ndarray can be a single row of numbers (1-D), a table of numbers (2-D), or even a cube of numbers (3-D and beyond). NumPy provides hundreds of functions to work on these arrays very quickly.

2.3.1 Why is NumPy Faster than Lists?

Python already has a built-in data structure called a list, so why do we need NumPy arrays? The answer is speed. Python lists can hold any kind of data, which makes them flexible but slow. NumPy arrays hold only one type of data (for example, all integers or all floating-point numbers). Because of this, NumPy can store the data efficiently in memory and use fast C code internally to perform calculations. The result is that NumPy is often 10 to 100 times faster than equivalent Python list operations.

2.3.2 Key Features of NumPy

N-Dimensional Arrays: NumPy provides the ndarray object, which can be a 1-D vector, 2-D matrix or higher-dimensional array. This is the basic building block for almost all numerical work.

Mathematical Functions: NumPy has built-in functions for arithmetic, trigonometry, logarithms, exponentials and many other mathematical operations. All these functions work directly on entire arrays, without writing loops.

Broadcasting: Broadcasting is a feature that lets us perform operations on arrays of different shapes without writing extra code. For example, we can add a single number to every element of an array in one step.

Random Number Generation: NumPy can generate random numbers from many different statistical distributions, which is useful for simulations and machine learning.

Linear Algebra: NumPy has functions for matrix multiplication, finding determinants, solving systems of equations, and decompositions like SVD and QR. These are essential for machine learning and statistics.

Memory Efficiency: NumPy arrays use much less memory than equivalent Python lists because the data is stored in a continuous block of memory of a single type.

Integration with C/C++ and Fortran: NumPy can call code written in C, C++ and Fortran, which makes it suitable for very high-performance computing.

2.3.3 A Simple NumPy Example

Let us see a small example. Suppose we want to find the average and the maximum of a list of marks of five students:

```
import numpy as np

marks = np.array([78, 85, 90, 65, 92])

print('Average marks:', np.mean(marks))
print('Maximum marks:', np.max(marks))
print('Minimum marks:', np.min(marks))
print('Standard deviation:', np.std(marks))
```

In this code, we first import NumPy as np. Then we create a NumPy array called marks. Finally, we use the built-in functions np.mean(), np.max(), np.min() and np.std() to calculate statistics.

Notice how short and clean the code is. Without NumPy, we would have had to write loops to calculate these values.

2.3.4 Advantages of NumPy

- Speed and Performance: NumPy operations are very fast because they use optimized C code internally.
- Memory Efficiency: Arrays use less memory than lists.
- Convenient Mathematical Operations: We can perform complex mathematical operations on entire arrays in one line.
- Compatibility: NumPy works with many other libraries like Pandas, Scikit-learn and Matplotlib.
- Versatile: It supports indexing, slicing, reshaping and many other operations on arrays.
- Foundation for Other Libraries: Almost every scientific computing library in Python uses NumPy underneath.

2.4 Pandas (Python Data Analysis)

Pandas is the most popular library for data analysis in Python. The name Pandas comes from "Panel Data", which is a term used in statistics for multi-dimensional structured data. Pandas is built on top of NumPy. While NumPy gives us fast arrays of numbers, Pandas gives us labelled tables of data, similar to what you see in Microsoft Excel or in a database table.

Pandas provides two main data structures: Series and DataFrame. A Series is a one-dimensional labelled array, like a single column of an Excel sheet. A DataFrame is a two-dimensional labelled table, like a complete Excel sheet. The DataFrame is the most commonly used data structure in Pandas. Almost all data science work in Python today is done using DataFrames.

2.4.1 Why Pandas?

Imagine you have a CSV file with the marks of all students in a college. The file has columns like Roll Number, Name, Branch, Semester and Marks. With Pandas, you can load this file into a DataFrame in one line of code. Then you can sort the data, filter rows, group by branch, calculate averages and create new columns, all very easily. Without Pandas, doing the same work would require many lines of complicated code.

2.4.2 Key Features of Pandas

DataFrame Object: The DataFrame is a two-dimensional labelled table. It can hold different types of data in different columns (numbers, text, dates, etc.) and lets us treat the data like a database table or an Excel sheet.

Series Object: A Series is a one-dimensional labelled array. It is the building block of a DataFrame. Each column of a DataFrame is actually a Series.

Data Manipulation: Pandas provides easy ways to filter rows, select columns, sort data, group data, pivot tables, merge two datasets and reshape data.

Handling Missing Data: In real-world data, some values are often missing. Pandas provides functions like isnull(), dropna() and fillna() to handle missing values easily.

Data Analysis Functions: Pandas has built-in functions for descriptive statistics, time series analysis, filtering and aggregation.

Connectivity with Other Libraries: Pandas works smoothly with NumPy, Matplotlib, Seaborn and Scikit-learn.

Input/Output Support: Pandas can read and write data in many file formats, including CSV, Excel, JSON, HTML and SQL databases.

2.4.3 A Simple Pandas Example

Let us see how Pandas can be used to work with student data:

```
import pandas as pd

# Create a DataFrame from a dictionary
data = {
   'Name': ['Aarav', 'Diya', 'Karan', 'Mira', 'Rohan'],
   'Branch': ['BCA', 'BCA', 'BSc IT', 'BCA', 'BSc IT'],
   'Marks': [78, 85, 90, 65, 92]
}
df = pd.DataFrame(data)

# Show the entire DataFrame
print(df)

# Show only students from BCA
bca_students = df[df['Branch'] == 'BCA']
```

```
print(bca_students)


# Average marks of each branch
print(df.groupby('Branch')['Marks'].mean())
```

In this example, we first create a DataFrame from a Python dictionary. Then we filter the DataFrame to get only BCA students using a condition. Finally, we group the data by Branch and calculate the average marks for each branch. With just three or four lines of code, we have done what would take many lines in plain Python.

2.4.4 Advantages of Pandas

- Easy to Use: The DataFrame structure feels natural to anyone who has worked with Excel.
- Handles Different Data Types: Numbers, text, dates and Boolean values can all be stored in the same DataFrame.
- Data Cleaning Tools: Provides easy ways to handle missing values, duplicates and inconsistent data.
- Powerful Group-by Operations: Allows us to group data by one or more columns and apply summary functions.
- Wide Range of File Formats: Reads and writes data from CSV, Excel, JSON, SQL and many other formats.
- Visualization Support: Built-in plotting functions that work directly with Matplotlib.
- Time Series Support: Special features for working with dates, times and time-based data.
- Scalable: Works well even with datasets containing hundreds of thousands of rows.

Check Your Progress - 1

What is a library in Python? Why do we use libraries instead of writing all the code ourselves?

Why are NumPy arrays faster than Python lists?

What is broadcasting in NumPy?

What are the two main data structures in Pandas? Briefly explain each.

Pandas is built on top of which library? (a) Matplotlib (b) NumPy (c) Scikit-learn (d) TensorFlow

Name any three file formats that Pandas can read.

2.5 Matplotlib

Matplotlib is the most widely used library for data visualization in Python. It was created by John D. Hunter in 2003. The name Matplotlib comes from "MATLAB plotting library", because it was designed to be similar to the plotting functions in MATLAB, a popular tool used by engineers and scientists.

With Matplotlib, we can create almost any kind of chart we can think of: line charts, bar charts, pie charts, scatter plots, histograms, box plots and many more. We can fully customize every part of the chart, including colours, labels, titles, axis ranges, legends and grid lines.

The most commonly used part of Matplotlib is its pyplot module. It provides a simple, MATLAB-like interface for creating charts. By convention, we usually import it as plt:

```
import matplotlib.pyplot as plt
```

2.5.1 A Simple Matplotlib Example

Let us draw a simple line chart showing the marks of a student in five subjects:

```
import matplotlib.pyplot as plt

subjects = ['Maths', 'English', 'Science', 'History', 'Computer']
marks = [78, 82, 90, 65, 95]

plt.plot(subjects, marks, marker='o')
plt.title('Marks of a BCA Student')
plt.xlabel('Subject')
plt.ylabel('Marks')
plt.grid(True)
plt.show()
```

This code will draw a line chart with the subject names on the x-axis and the marks on the y-axis. The marker='o' option adds a circle at each data point. plt.show() is the command that actually displays the chart on the screen. With just six or seven lines, we get a complete, professional-looking chart.

2.5.2 Key Features of Matplotlib

Many Types of Plots: Matplotlib supports line plots, bar charts, scatter plots, pie charts, histograms, box plots, error bars and many more.

Full Customization: Almost every element of a chart, including colours, line styles, fonts, labels and legends, can be customized.

Multiple Plots in One Figure: We can create several charts side by side in one figure, which is helpful for comparing different things.

3D Plotting: Matplotlib supports basic 3D charts using its mplot3d toolkit.

Animation: We can create animated charts that show changes over time.

Save in Many Formats: Charts can be saved as PNG, JPG, PDF, SVG and EPS files.

Works with Other Libraries: Matplotlib works seamlessly with NumPy, Pandas and Seaborn.

Jupyter Notebook Integration: Charts appear directly inside Jupyter notebooks, which is great for interactive analysis.

2.5.3 Advantages of Matplotlib

- Easy for Beginners: The pyplot module has a simple, function-based interface.
- Highly Customizable: Experts can fine-tune every aspect of a chart for publication-quality output.
- Wide Variety of Charts: Supports almost every type of standard chart.
- Free and Open Source: No license fees, even for commercial use.
- Large Community and Documentation: There are tutorials and examples for nearly every kind of chart.

2.6 Seaborn

Seaborn is another important visualization library in Python. It is built on top of Matplotlib but is designed to make statistical charts much easier and more attractive. While Matplotlib lets us draw any kind of chart, Seaborn focuses on the most common types of statistical charts and makes them look beautiful with very little code.

Seaborn was created by Michael Waskom. The name has no special meaning; it was just a unique word the author chose. Seaborn comes with built-in themes (like darkgrid, whitegrid, ticks) and colour palettes that make our charts look professional automatically.

By convention, Seaborn is usually imported as sns:

```
import seaborn as sns
```

2.6.1 A Simple Seaborn Example

Suppose we have data about the daily sales of a small shop and we want to see the relationship between the temperature and ice-cream sales. Seaborn can draw a scatter plot with a regression line in just one command:

```
import seaborn as sns
import matplotlib.pyplot as plt

data = sns.load_dataset('tips')  # Built-in sample dataset
sns.scatterplot(data=data, x='total_bill', y='tip')
plt.title('Tip vs Total Bill')
plt.show()
```

In this example, we use Seaborn's built-in tips dataset, which contains data about restaurant tips. We then draw a scatter plot of the tip amount against the total bill. Seaborn automatically picks nice colours and styles.

2.6.2 Key Features of Seaborn

Built-in Themes and Colour Palettes: Seaborn provides several beautiful themes and colour palettes that we can apply with one command.

Statistical Plots: It has special functions for distribution plots, regression plots, categorical plots, heatmaps and pair plots.

Easy Pandas Integration: Seaborn works directly with Pandas DataFrames. We just pass the DataFrame and the column names.

Automatic Aggregation: When making bar plots or box plots from raw data, Seaborn calculates the necessary statistics (means, medians, etc.) automatically.

FacetGrid: We can create a grid of small charts based on different categories of the data, which is very useful for comparing groups.

2.6.3 Common Seaborn Chart Types

- histplot(): For drawing histograms.
- scatterplot(): For drawing scatter plots.
- lineplot(): For line charts.
- barplot(): For bar charts.
- boxplot(): For box plots.
- heatmap(): For drawing colour-coded matrices, often used for correlation.
- pairplot(): For drawing scatter plots between every pair of columns.
- regplot(): For scatter plots with a regression line.

2.6.4 Difference between Matplotlib and Seaborn

Both Matplotlib and Seaborn are used for visualization, so what is the difference? The simple answer is that Matplotlib gives us full control but needs more code, while Seaborn gives us

beautiful charts with less code but less control. Most data scientists use both: they use Seaborn for quick, attractive charts during exploration, and Matplotlib for fine-tuning when they need a specific look.

2.7 Scikit-learn

Scikit-learn (sometimes written as sklearn) is the most popular machine learning library in Python. It was first released in 2007 and has become the standard tool for classical machine learning. Scikit-learn is built on top of NumPy, SciPy and Matplotlib. It provides simple and efficient tools for almost every standard machine learning task.

As a BCA student new to machine learning, you can think of Scikit-learn as a giant toolbox. Every tool in this box is a different machine learning algorithm. To use a tool, you simply pick it from the box, fit it to your data, and then ask it to make predictions on new data. The interface is very consistent, which means once you learn how to use one algorithm, you can use almost any other algorithm in the same way.

2.7.1 What Can Scikit-learn Do?

Scikit-learn supports four main types of machine learning tasks.

Classification: Predicting a category. For example, predicting whether an email is spam or not, or whether a student will pass or fail.

Regression: Predicting a numerical value. For example, predicting the price of a house, or the marks a student will get.

Clustering: Grouping similar items together. For example, grouping customers based on their buying behaviour.

Dimensionality Reduction: Reducing the number of variables in the data while keeping most of the information. This is useful when the data has too many columns.

2.7.2 Key Features of Scikit-learn

Many Algorithms: It includes algorithms for classification (logistic regression, decision trees, random forests, support vector machines, k-nearest neighbours, naive Bayes), regression (linear regression, ridge, lasso), clustering (k-means, DBSCAN, hierarchical) and dimensionality reduction (PCA, t-SNE).

Pre-processing Tools: Scikit-learn provides functions for scaling features, encoding categorical variables, handling missing values and creating polynomial features.

Model Selection: It has tools for splitting data into training and testing sets, performing cross-validation, and tuning model parameters automatically.

Evaluation Metrics: Built-in functions for accuracy, precision, recall, F1-score, ROC-AUC, mean squared error and many more.

Pipelines: We can create a pipeline that combines pre-processing and model fitting into a single object. This makes the code cleaner and prevents data leakage.

Consistent API: Almost all algorithms follow the same pattern: create the model, call fit() to train it, and call predict() to get predictions.

2.7.3 A Simple Scikit-learn Example

Let us see a simple example where we use Scikit-learn to build a model that predicts whether a student will pass based on the number of hours they study:

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
import numpy as np

# Sample data: hours studied and pass/fail
hours = np.array([[1], [2], [3], [4], [5], [6], [7], [8]])
result = np.array([0, 0, 0, 0, 1, 1, 1, 1])  # 1 = pass, 0 = fail

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    hours, result, test_size=0.25)

# Create and train the model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions
prediction = model.predict([[5]])
print('Will the student pass?', prediction)
```

In this small example, we have used logistic regression to predict the result. The model learns from the training data and then predicts whether a student who studies for 5 hours will pass or fail. With Scikit-learn, the entire machine learning workflow can be done in just a few lines.

2.8 Other Useful Libraries

Apart from the five main libraries we have studied, there are several other libraries that are useful in data science. Let us look at them briefly.

2.8.1 SciPy

SciPy stands for Scientific Python. It is built on top of NumPy and provides additional functions for scientific computing. SciPy includes modules for optimization, integration, interpolation, signal processing, image processing, and more advanced statistics. While NumPy gives us the basic numerical tools, SciPy gives us the advanced ones.

2.8.2 Statsmodels

Statsmodels is a Python library focused on statistical modelling. It provides classes and functions for many statistical techniques, including linear regression, time series analysis, hypothesis testing and survival analysis. While Scikit-learn is more focused on prediction, Statsmodels is more focused on statistical inference and provides detailed output similar to what one would see in statistical software like SPSS or SAS.

2.8.3 TensorFlow and Keras

TensorFlow is an open-source library developed by Google for deep learning. Deep learning is a special type of machine learning that uses neural networks with many layers. TensorFlow is used for tasks like image recognition, voice recognition and language translation. Keras is a high-level interface that makes TensorFlow easier to use. Keras is now part of TensorFlow.

2.8.4 PyTorch

PyTorch is another popular deep learning library, developed by Meta (Facebook). It is widely used in research and is becoming popular in industry as well.

2.8.5 Plotly

Plotly is a library for creating interactive charts. While Matplotlib and Seaborn make static charts (charts that do not change), Plotly charts can be zoomed, panned and explored. Plotly charts work very well in web applications and dashboards.

2.8.6 BeautifulSoup and Scrapy

These two libraries are used for web scraping, which is the process of extracting data from websites. BeautifulSoup is simpler and good for small projects. Scrapy is a more powerful framework for large-scale web scraping projects.

2.9 How These Libraries Work Together

In a real data science project, we usually use several libraries together. Let us see a typical workflow.

1. Use Pandas to load the data from a CSV file or database.
2. Use Pandas and NumPy to clean the data and handle missing values.
3. Use Pandas, Matplotlib and Seaborn to explore the data and find patterns.
4. Use Scikit-learn to pre-process the data (scaling, encoding, etc.).
5. Use Scikit-learn to build a machine learning model and evaluate it.
6. Use Matplotlib or Plotly to visualize the results.
7. Use Flask, Streamlit or Dash to deploy the model as a web application.

This pipeline is followed in almost every data science project. Once you become comfortable with the five main libraries (NumPy, Pandas, Matplotlib, Seaborn and Scikit-learn), you can solve a very wide range of data science problems.

Check Your Progress - 2

What does the name Matplotlib stand for?

Name any four types of charts that we can create using Matplotlib.

Seaborn is built on top of which library?

0. Why do we use Seaborn instead of plain Matplotlib for some charts?
1. Which Python library is most commonly used for machine learning? List four types of tasks it supports.
2. Match the library with its main use: (i) NumPy (ii) Pandas (iii) Matplotlib (iv) Scikit-learn — (a) Machine learning, (b) Numerical arrays, (c) Data tables, (d) Charts.

2.10 Let us sum up

In this unit, we have studied the most important Python libraries used in data science. We started by understanding what a library is and why we use libraries. We then studied NumPy, which gives us fast multidimensional arrays and is the foundation of almost every other data science library in Python. Next, we looked at Pandas, which is built on top of NumPy and provides DataFrames, the most popular data structure in data science. After Pandas, we studied Matplotlib, the most widely used Python library for drawing charts. We learned about its pyplot module and saw a simple example of drawing a line chart.

We then studied Seaborn, which is built on top of Matplotlib and makes statistical charts very easy and attractive. We compared Matplotlib and Seaborn and noted that Seaborn is best for quick, attractive charts while Matplotlib gives more control. After Seaborn, we studied Scikit-learn, which is the most popular machine learning library in Python. We saw the four main types of tasks it supports: classification, regression, clustering and dimensionality reduction.

2.11 Check your progress: Possible Answers

A library in Python is a collection of pre-written functions and classes that we can use in our own programs. We use libraries because they save time, are well-tested and reliable, are optimized for speed, follow standard ways of doing things, and provide a wide range of features.

NumPy arrays are faster than Python lists because (a) NumPy arrays store all elements of the same data type in a single contiguous block of memory, (b) NumPy uses optimized C code internally, and (c) NumPy can perform operations on entire arrays at once without writing Python loops.

Broadcasting is a feature of NumPy that allows us to perform operations on arrays of different shapes without manually expanding them. For example, we can add a single number to every element of an array, or add a 1-D array to every row of a 2-D array, in one step.

The two main data structures in Pandas are Series and DataFrame. A Series is a one-dimensional labelled array, similar to a single column of an Excel sheet. A DataFrame is a two-dimensional labelled table with rows and columns, similar to a complete Excel sheet or a database table.

(b) NumPy.

Three file formats Pandas can read: CSV, Excel and JSON. (Other valid answers include SQL databases, HTML and Parquet.)

The name Matplotlib stands for "MATLAB plotting library", because it was designed to be similar to the plotting functions in MATLAB.

Four types of charts we can create with Matplotlib: line charts, bar charts, scatter plots and histograms. (Other valid answers include pie charts, box plots, area charts, etc.)

Seaborn is built on top of Matplotlib.

We use Seaborn instead of plain Matplotlib for some charts because Seaborn produces more attractive statistical charts with less code, has built-in themes and colour palettes, works directly with Pandas DataFrames, and automatically performs aggregation when needed.

Scikit-learn is the most commonly used Python library for machine learning. The four main types of tasks it supports are classification, regression, clustering and dimensionality reduction.

(i) NumPy - (b) Numerical arrays; (ii) Pandas - (c) Data tables; (iii) Matplotlib - (d) Charts; (iv) Scikit-learn - (a) Machine learning.

2.12 Further Readings

1. Wes McKinney, "Python for Data Analysis", O'Reilly Media, 3rd Edition, 2022.
2. Jake VanderPlas, "Python Data Science Handbook", O'Reilly Media, 2nd Edition, 2023.
3. Aurelien Geron, "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow", O'Reilly Media, 3rd Edition, 2022.
4. Anurag Gupta and G. P. Biswas, "Python Programming: Problem Solving, Packages and Libraries", McGraw Hill, 2019.
5. Online resource: Pandas official documentation at https://pandas.pydata.org/
6. Online resource: Matplotlib gallery at https://matplotlib.org/stable/gallery/
7. Online resource: Scikit-learn user guide at https://scikit-learn.org/

2.13 Assignments

1. What is a Python library? Explain why libraries are very useful in data science. Give any five general features of Python libraries used in data science.
2. Explain the key features of the NumPy library. Why are NumPy arrays preferred over Python lists for numerical work?
3. What is a DataFrame in Pandas? Write a simple Python program using Pandas to (a) create a DataFrame of any 5 students with columns Name, Branch and Marks, (b) print all students from the BCA branch, and (c) calculate the average marks.
4. Explain the role of Matplotlib in data science. List any five types of charts we can create using Matplotlib.
5. Compare Matplotlib and Seaborn. When would you prefer Seaborn over Matplotlib?
6. What is Scikit-learn? Describe the four main types of machine learning tasks that Scikit-learn supports.
7. Briefly describe the role of any four other libraries (apart from NumPy, Pandas, Matplotlib, Seaborn and Scikit-learn) that are useful in data science.\
8. Describe a typical data science workflow and explain at which step each of the following libraries is used: Pandas, NumPy, Matplotlib, Seaborn and Scikit-learn.

Unit-3: Working with Jupyter Notebooks

Unit Structure

3.0 Learning Objectives

After studying this unit student should be able to:

- Understand what a Jupyter Notebook is and why it is so popular in data science.
- Install Jupyter Notebook on a personal computer using Anaconda or pip.
- Navigate the Jupyter Notebook interface and understand its main components.
- Create and use different types of cells: Code, Markdown and Raw.
- Run code, save notebooks and export them in different formats.
- Use keyboard shortcuts to work faster in Jupyter.
- Customize Jupyter Notebooks with themes and extensions.
- Follow best practices for writing clear and readable notebooks.

3.1 Introduction to Jupyter Notebooks

In data science, we often want to do things step by step. We may load some data, look at it, then clean a few columns, then look at it again, then make a chart, and so on. We also want to keep notes about what we are doing and why. We want our charts to appear next to the code that creates them. We want to share our work with others in a way that lets them see our code, our explanations and our results, all in one document. This is exactly what Jupyter Notebooks let us do.

Jupyter Notebook is an open-source web application that allows us to create and share documents containing live code, equations, charts and explanatory text. The name Jupyter is a combination of the three core programming languages it supports: Julia, Python and R. Although it was originally built mainly for Python, Jupyter today supports more than 40 programming languages.

A Jupyter Notebook is saved as a file with the extension .ipynb (which stands for IPython Notebook, the original name of the project). The .ipynb file is actually a special JSON file that stores all the cells, the code, the output and the markdown text. We can share this file with anyone, and if they have Jupyter installed, they can open it and run our code.

3.2 Why Jupyter Notebooks?

Jupyter Notebooks have become so popular in data science because they offer many benefits. Let us look at the main features and advantages.

1. Interactivity: In Jupyter, we write our code in small blocks called cells. We can run any cell at any time and see the output immediately below the cell. This makes it easy to try things, see

what happens and modify our approach. This kind of step-by-step, interactive style of programming is perfect for data analysis.

2. Documentation Alongside Code: In a Jupyter Notebook, we can have cells of different types. Code cells contain code and show the output. Markdown cells contain formatted text, including headings, bullet points, images and even mathematical equations. We can place explanation cells next to code cells, which makes the notebook a complete document, not just a program.

3. Inline Visualizations: Charts created with libraries like Matplotlib and Seaborn appear directly inside the notebook. We can see our chart right next to the code that produced it. This is very helpful when exploring data.

4. Easy to Share: A notebook saved as a .ipynb file can be shared by email or uploaded to GitHub, where it will be displayed nicely in the browser. We can also export the notebook to HTML or PDF and send it as a regular document.

5. Support for Multiple Languages: Although Python is the most common, Jupyter also supports R, Julia, Scala and many other languages. We choose the language by selecting an appropriate "kernel". A kernel is a small program that runs in the background and executes the code in the notebook.

6. Extensibility: Jupyter supports many extensions that add useful features such as a table of contents, code folding, spell checker and many others.

7. Integration with Cloud Platforms: Online services like Google Colab, Kaggle Notebooks and Microsoft Azure Notebooks allow us to run Jupyter Notebooks directly in a web browser without installing anything. This is very useful for students who do not have a powerful computer.

8. Reproducible Research: Because the notebook contains everything (code, output, explanations), anyone who reads it can reproduce the results by simply running the cells again. This is very important in research.

3.3 Installation and Setup

Now let us see how to install Jupyter Notebook on a personal computer. There are three common ways to do it.

3.3.1 Method 1: Using Anaconda (Recommended)

- Anaconda is a free Python distribution that comes with hundreds of pre-installed libraries for data science, including Jupyter Notebook. For most students, this is the easiest and best way to start. The steps are as follows.
- Open a web browser and go to https://www.anaconda.com/download.

- Click on the download link for your operating system (Windows, macOS or Linux).
- Once the installer is downloaded, double-click on it to start the installation.
- Follow the on-screen instructions. When asked, select the option to add Anaconda to your system PATH (this makes it easier to use later).
- After installation is complete, open the Anaconda Navigator from the Start menu (on Windows) or the Applications folder (on macOS).
- In the Navigator, you will see several applications, including Jupyter Notebook and JupyterLab. Click on "Launch" under Jupyter Notebook.
- Jupyter Notebook will open in your default web browser. You are now ready to use it.
- Anaconda is recommended because it comes with all the popular data science libraries already installed. You will not have to install Pandas, NumPy, Matplotlib or Scikit-learn separately.

3.3.2 Method 2: Installing Jupyter using pip

- If you already have Python installed and you do not want to install Anaconda, you can install Jupyter Notebook using pip. The steps are as follows.
- Open a command prompt (on Windows) or a terminal (on macOS or Linux).
- Type the following command and press Enter:
- pip install notebook
- Wait for the installation to complete. It may take a few minutes.
- Once installed, you can start Jupyter Notebook by typing the following command:
- jupyter notebook
- Jupyter Notebook will open in your default web browser.
- If you also want JupyterLab, which is a more modern interface, you can install it with:
- pip install jupyterlab
- And start it with:
- jupyter lab

3.3.3 Method 3: Online Notebooks (No Installation)

- If you do not want to install anything, you can use Jupyter Notebooks directly in your web browser through online services. Some popular options are listed below.
- Google Colab (https://colab.research.google.com): A free service from Google that lets us run Python notebooks in the cloud. It also provides free GPU access for machine learning.

- Kaggle Notebooks (https://www.kaggle.com): A platform popular among data science learners. It hosts many datasets and free notebooks.
- Microsoft Azure Notebooks: A cloud-based notebook service from Microsoft.
- Binder (https://mybinder.org): A free service that turns a GitHub repository into a runnable Jupyter Notebook environment.
- These online services are very useful for BCA students because they do not require any installation and they work even on slow computers. For your initial learning, Google Colab is highly recommended.

3.3.4 What is JupyterLab?

JupyterLab is the next-generation interface for Project Jupyter. It is more modern and more flexible than the classic Jupyter Notebook. With JupyterLab, we can have multiple notebooks, terminals and text editors open side by side, like in a regular code editor. JupyterLab is included in Anaconda and can also be installed separately using pip. As a beginner, you can use either Jupyter Notebook (classic) or JupyterLab (modern) — both work well.

3.4 Navigating the Jupyter Notebook Interface

When you launch Jupyter Notebook, the first thing you see in your web browser is the Notebook Dashboard. This is the home screen where you can see all the files and folders in the current directory, create new notebooks and open existing ones.

3.4.1 Notebook Dashboard

The Notebook Dashboard has three main tabs.

- Files Tab: Shows the files and folders in the current directory. You can click on any .ipynb file to open it as a notebook.
- Running Tab: Shows the notebooks that are currently running. You can stop any running notebook from here.
- Clusters Tab: Used for managing parallel computing clusters. As a beginner, you do not need to worry about this.
- On the right side, there is a New button. Clicking on it gives us options to create a new notebook (Python 3, R, etc.), a new text file or a new folder.

3.4.2 The Notebook Itself

When you open or create a notebook, you see the actual notebook interface. It has the following parts.

1. Notebook Name: At the very top, you see the name of the notebook. By default, it is called "Untitled". Click on it to change the name.
2. Menu Bar: Just below the name, the menu bar contains menus like File, Edit, View, Insert, Cell, Kernel, Help. Each menu has many options.
3. Toolbar: Below the menu bar, the toolbar has icons for common actions like Save, Add Cell, Cut, Copy, Paste, Move Up, Move Down, Run, Stop and Restart. There is also a dropdown to change the cell type.
4. Cells Area: This is the main working area where you write your code and your text. Each cell is a separate block.
5. Kernel Indicator: In the upper right corner, there is a small circle. When the circle is empty, the kernel is idle (not running anything). When the circle is filled, the kernel is busy (running your code). Next to the circle, the name of the kernel is shown (for example, Python 3).

3.5 Working with Cells: Code, Markdown and Raws

Cells are the building blocks of a Jupyter Notebook. There are three main types of cells, each used for a different purpose.

3.5.1 Code Cells

Code cells contain code that can be run by the kernel. The output of the code (text, tables, charts) appears just below the cell. By default, every new cell is a code cell. The language of the code depends on the kernel chosen for the notebook (usually Python).

To run a code cell, click on it and then press Shift + Enter, or click the Run button in the toolbar. The output appears below the cell. While the cell is running, you will see In [*] on the left side of the cell. When it is done, the * is replaced by a number, like In [1], In [2] and so on.

Here is an example of a code cell:

```
# This is a code cell
name = 'Sanskruti'
course = 'BCA Data Science'
print(f'Welcome to {course}, {name}!')
```

3.5.2 Markdown Cells

Markdown cells are used for writing formatted text. Markdown is a simple language for adding formatting to plain text using a few special symbols. For example, putting a hash (#) before a line makes it a heading. Putting two stars (**) around a word makes it bold.

Markdown cells are very useful in data science notebooks. We use them to write the title, the introduction, the explanation of each step, the conclusion, and so on. They make our notebook a proper document that anyone can read and understand.

Here are some examples of Markdown formatting:

```
# Main Heading
## Sub Heading
### Smaller Heading

This is a paragraph with **bold text** and *italic text*.
- This is a bullet point
- Another bullet point

1. This is a numbered list
2. Second item

[Click here](https://example.com) is a hyperlink.

`code in line` and triple-backtick code block.
```

To create a Markdown cell, first create a new cell (it will be a code cell by default), then change its type to Markdown using the dropdown in the toolbar, or by pressing M in command mode.

3.5.3 Raw Cells

Raw cells display content as it is, without any formatting or execution. They are not used very often. Their main purpose is to include text that should not be processed when the notebook is exported. As a beginner, you will rarely need raw cells.

3.5.4 Adding and Modifying Cells

There are several ways to add a new cell.

1. Click the + icon in the toolbar.
2. Use the Insert menu to insert a cell above or below the current cell.
3. Press A in command mode to insert a cell above.
4. Press B in command mode to insert a cell below.
5. To change the type of a cell, click on the cell and use the dropdown in the toolbar. Alternatively, press M in command mode to change to Markdown, Y to change to Code, or R to change to Raw

3.6 Essential Operations

3.6.1 Running Code

There are several ways to run a cell.

1. Press Shift + Enter to run the cell and move to the next cell.
2. Press Ctrl + Enter to run the cell and stay on the same cell.
3. Press Alt + Enter to run the cell and insert a new cell below.
4. Click the Run button in the toolbar.

If you have a long-running cell and want to stop it, click the Stop button in the toolbar (it looks like a square). If something goes very wrong (for example, the kernel becomes unresponsive), you can restart the kernel from the Kernel menu. Restarting will clear all the variables and outputs but will not delete your code.

3.6.2 Saving Work

Jupyter Notebook automatically saves your work every two minutes by default. However, it is a good habit to save manually too. You can save by pressing Ctrl + S (Cmd + S on macOS) or by clicking the Save icon (a floppy disk) in the toolbar. Each save creates a checkpoint, which is a snapshot of the notebook. If you make a mistake, you can revert to the last checkpoint from the File menu.

3.6.3 Exporting Notebooks

Jupyter Notebooks can be exported to several formats for sharing or printing.

1. HTML: A web page with all the code, output and text. Anyone can open this in a browser without needing Jupyter.
2. PDF: A printable document. (You may need to install LaTeX for this to work.)
3. Python (.py): A plain Python script with only the code (no output, no Markdown).
4. Markdown (.md): A plain Markdown document.
5. LaTeX (.tex): For producing high-quality printed documents.
6. To export, go to File > Download as and select the desired format.

Check your progress-1

What is a Jupyter Notebook? Why is the file extension .ipynb?

Name any three programming languages supported by Jupyter Notebook.

Why is Anaconda the recommended way to install Jupyter Notebook for beginners?

What is Google Colab? Give one advantage of using it.

Fill in the blank: A _____ in Jupyter is a small program that runs in the background and executes our code.

What is the difference between a Code cell and a Markdown cell?

3.7 Keyboard Shortcuts

Once you become comfortable with Jupyter Notebook, you will find that using the mouse for everything slows you down. Keyboard shortcuts help you work much faster. Jupyter has two modes that change how the keyboard works.

3.7.1 Command Mode and Edit Mode

Jupyter has two distinct modes.

Edit Mode: When you click inside a cell to type, you are in Edit Mode. In this mode, the keyboard works normally, just like in any text editor. You can type code or text. The cell border is green when you are in Edit Mode.

Command Mode: When you press the Esc key, you exit Edit Mode and enter Command Mode. In this mode, the keyboard keys are used for shortcuts that work on cells (like adding, deleting, copying cells). The cell border is blue when you are in Command Mode.

To go back to Edit Mode from Command Mode, simply press Enter or click inside a cell.

3.7.2 List of Useful Keyboard Shortcuts

The table below lists the most useful keyboard shortcuts in Jupyter Notebook. Try to memorize and use them. They will save you a lot of time

Mode	Shortcut	Action	Description
Command	A	Insert above	Add a new cell above the current cell
Command	B	Insert below	Add a new cell below the current cell
Command	M	To Markdown	Convert the current cell to a Markdown cell
Command	Y	To Code	Convert the current cell to a Code cell
Command	D, D	Delete cell	Press D twice to delete the current cell
Command	Z	Undo delete	Restore the most recently deleted cell

Mode	Shortcut	Action	Description
Command	X	Cut cell	Cut the current cell
Command	C	Copy cell	Copy the current cell
Command	V	Paste cell	Paste the copied cell below the current cell
Edit	Shift + Enter	Run and move	Run the current cell and select the next cell
Edit	Ctrl + Enter	Run and stay	Run the current cell and stay on the same cell
Edit	Alt + Enter	Run and insert	Run the cell and insert a new cell below
Edit	Tab	Code complete	Show suggestions for completing code
Edit	Shift + Tab	Show help	Show documentation for the function under cursor

To see the complete list of keyboard shortcuts, go to the Help menu in Jupyter and click on Keyboard Shortcuts. You can even customize the shortcuts to suit your preference.

3.8 Customizing Themes and Extensions

Jupyter Notebook can be customized in many ways to make it more comfortable and powerful. The two main ways are themes and extensions.

3.8.1 Themes

By default, Jupyter Notebook has a white background. Many people prefer a dark background, especially when working at night, because it is easier on the eyes. We can change the theme using a package called jupyterthemes.

To install jupyterthemes, run the following command in your terminal or command prompt:

```
pip install jupyterthemes
```

After installation, you can apply a theme using:

```
jt -t monokai
```

This applies the monokai theme, which is a popular dark theme. Other available themes include onedork, grade3, oceans16, solarizedl, solarizedd, gruvboxl and gruvboxd. To see the list of available themes, run:

```
jt -l
```

To go back to the default theme, run:

```
jt -r
```

3.8.2 Extensions

Extensions add new features to Jupyter Notebook. They can be installed using a package called jupyter_contrib_nbextensions.

To install it, run:

```
pip install jupyter_contrib_nbextensions
jupyter contrib nbextension install --user
```

After installation, you can manage extensions in your browser at http://localhost:8888/nbextensions when Jupyter is running. Some useful extensions are listed below.

1. Table of Contents: Adds an automatic table of contents to your notebook based on the Markdown headings.
2. Code Folding: Lets you hide and show parts of your code, useful for long cells.
3. Execute Time: Shows how long each cell took to run.
4. Variable Inspector: Shows all the variables and their values in a side panel.
5. Spellchecker: Underlines spelling mistakes in Markdown cells.
6. Codefolding: Lets you fold (collapse) and unfold blocks of code.
7. These extensions can make your work much more efficient. As a BCA student, try installing a few of them and see how they help.

3.9 Best Practices for Writing Good Notebooks

A Jupyter Notebook is more than just a piece of code. It is a document that tells a story. To make our notebooks useful and easy to understand, we should follow some good practices.

- Give Your Notebook a Clear Name: Avoid names like "Untitled" or "Test". Use descriptive names like "Customer Sales Analysis - January 2026".
- Start with a Title and Introduction: The first cell should be a Markdown cell with the title of the notebook, your name, the date, and a short description of what the notebook does.

- Use Markdown Cells Generously: Add Markdown cells to explain the goal of each section, the steps you are taking and the conclusions you are drawing. A notebook without explanations is hard to understand later.
- Keep Code Cells Short: Each code cell should do one specific thing. Long cells with many tasks are hard to debug. If a cell is doing many things, split it into smaller cells.
- Use Comments in Code: Even within a code cell, use # comments to explain what each part does.
- Run Cells in Order: Always run the cells in order from top to bottom. Running cells out of order can cause confusing results because the order of execution affects the values of variables.
- Restart and Run All Before Sharing: Before sharing a notebook, click Kernel > Restart and Run All. This makes sure that all cells run correctly from a clean state. If something fails, you can fix it before sharing.
- Clear Output Before Saving (when appropriate): Sometimes, the output of cells (especially large tables or charts) makes the .ipynb file very big. You can clear all outputs from Cell > All Output > Clear before saving, if needed.
- Use Consistent Naming: Use clear and consistent names for variables, columns and functions. For example, use student_marks instead of x or sm.
- Save Your Work Often: Use Ctrl + S regularly. Power cuts and crashes can happen, and you do not want to lose your work

Check Your Progress - 2

1. What is the difference between Command Mode and Edit Mode in Jupyter Notebook?
2. Which keyboard shortcut is used to: (a) Run a cell and move to the next, (b) Run a cell and stay on it, (c) Insert a new cell below, (d) Delete a cell?
3. What does the package jupyterthemes do?
4. Name any three useful Jupyter Notebook extensions.
5. Why is it a good practice to click "Restart and Run All" before sharing a notebook?
6. Fill in the blank: To convert a Code cell to a Markdown cell in Command Mode, press the key _____.

3.8 Let us sum up

In this unit, we have studied Jupyter Notebooks, which are the most popular working environment for data scientists. We started by understanding what a Jupyter Notebook is and why it is so widely used. We then learned about three different ways to install Jupyter Notebook on our computer: using Anaconda (recommended for beginners), using pip, and using online services like Google Colab (which require no installation at all). After installation, we explored the Jupyter Notebook interface, including the Notebook Dashboard, the menu bar, the toolbar, the cells area and the kernel indicator. We then studied the three types of cells: Code, Markdown and Raw, and saw simple examples of each. We covered essential operations such as running code, saving work and exporting notebooks to different formats. We then learned about Command Mode and Edit Mode, and listed many useful keyboard shortcuts that help us work faster. By the end of this unit, you should be comfortable installing, opening, using and sharing Jupyter Notebooks. In the next unit, we will use Jupyter to actually do data analysis.

3.9 Check your progress: Possible Answers

A Jupyter Notebook is a web-based document that combines live code, formatted text, equations, charts and other rich media in a single file. It allows interactive computing where we can run code in small blocks (cells) and see the results immediately. The file extension is .ipynb because Jupyter was originally called IPython Notebook (Interactive Python Notebook).

Three programming languages supported by Jupyter Notebook: Python, R and Julia. (The name Jupyter itself comes from these three languages.)

Anaconda is the recommended way to install Jupyter Notebook for beginners because it is a complete distribution that already includes Python, Jupyter Notebook, and most popular data science libraries (NumPy, Pandas, Matplotlib, Scikit-learn, etc.). It also provides Anaconda Navigator, a graphical tool to manage environments and launch applications. With Anaconda, beginners can start working immediately without installing libraries one by one.

Google Colab is a free online service from Google that lets us run Jupyter Notebooks in the cloud through a web browser. One advantage is that we do not need to install anything on our computer. Other advantages include free GPU access for machine learning, easy sharing through Google Drive, and good collaboration features.

A kernel is a small program that runs in the background and executes our code.

A Code cell contains code that is executed by the kernel, and its output appears just below the cell. A Markdown cell contains formatted text using Markdown syntax (headings, bullets, links, etc.) and is used for explanations and documentation. Markdown cells are not executed; they are only rendered as formatted text.

Command Mode is used to manage cells (insert, delete, move, change type) and is entered by pressing Esc. The cell border is blue. Edit Mode is used to type and edit the content of a cell, and is entered by pressing Enter or clicking inside a cell. The cell border is green.

Keyboard shortcuts: (a) Shift + Enter to run a cell and move to the next, (b) Ctrl + Enter to run a cell and stay on it, (c) B (in Command Mode) to insert a new cell below, (d) D, D (press D twice in Command Mode) to delete a cell.

The package jupyterthemes lets us change the visual theme (colours, fonts, background) of the Jupyter Notebook interface. It provides several built-in themes such as monokai, onedork and solarized, and a popular use is to switch from the default white background to a dark background for comfortable use at night.

Three useful Jupyter Notebook extensions: Table of Contents, Code Folding and Execute Time. (Other valid answers include Variable Inspector, Spellchecker, Codefolding, etc.)

It is good practice to "Restart and Run All" before sharing a notebook because it ensures that the notebook can run correctly from start to finish in a clean state. Sometimes during development we run cells out of order, which can hide errors. Restart and Run All catches such issues so that the recipient gets a notebook that just works.

To convert a Code cell to a Markdown cell in Command Mode, press the key M.

3.10 Further Readings

1. Kennedy Behrman, "Foundational Python for Data Science", Pearson, 1st Edition, 2022.
2. Marc Wintjen and Andrew Vlahutin, "Practical Data Analysis Using Jupyter Notebook", Packt Publishing, 2020.
3. Online resource: Official Jupyter documentation at https://docs.jupyter.org/
4. Online resource: Anaconda documentation at https://docs.anaconda.com/
5. Online resource: Google Colab quick guide at https://colab.research.google.com/notebooks/intro.ipynb

6. Online resource: Jupyter Notebook tutorial at https://realpython.com/jupyter-notebook-introduction/

3.11 Assignments

1. What is a Jupyter Notebook? Explain at least five features that make Jupyter Notebooks ideal for data science work.
2. Compare the three methods of installing Jupyter Notebook (Anaconda, pip, online services). Which method would you recommend to a fellow BCA student and why?
3. Describe the main components of the Jupyter Notebook interface. What is the role of the kernel indicator?
4. Explain the three types of cells in Jupyter Notebook. Give one example of when you would use each type.
5. Make a list of at least 10 keyboard shortcuts in Jupyter Notebook and explain what each does.
6. Install Jupyter Notebook on your computer (using either Anaconda or pip). Create a new notebook, write a small Python program that prints your name and current branch, and save the notebook. Take a screenshot and submit it.
7. Install the jupyterthemes package and apply a dark theme of your choice. Then revert to the default theme. Describe each step you followed.
8. Install at least three Jupyter Notebook extensions. Explain how each extension helps you work better.

Unit-4: Data Handling and Analysis with Jupyter Notebooks

Unit Structure

4.0 Learning Objectives

After studying this unit, the student should be able to:

- Import data into a Jupyter Notebook from CSV, Excel, JSON, SQL and APIs.
- Use Pandas and NumPy to load and manipulate data.
- Import data from both local files and remote URLs.
- Handle large datasets using chunking and sampling techniques.
- Explore data using summary statistics and identify different data types.
- Identify and handle missing values and duplicate rows.
- Transform and format data, including type conversion, normalization and feature engineering.
- Create basic visualizations like histograms, box plots and scatter plots for exploratory data analysis.
- Perform a complete data exploration on a small dataset from start to finish.

4.1 Introduction

In the previous unit, we learned how to install and use Jupyter Notebooks. Now we will use Jupyter to do real data analysis work. This unit covers the most important steps that come after we open Jupyter and before we build any model. These steps include importing data, exploring it, cleaning it and transforming it. Together, these steps are often called "data wrangling" or "data preparation". They usually take 70 to 80 percent of the total time of a data science project.
To make these concepts concrete, we will use small Python code examples throughout this unit. You should try every example yourself in your Jupyter Notebook. The best way to learn data analysis is by doing it. Reading is good, but writing the code yourself is much better.
By the end of this unit, you will be able to take a CSV file (for example, a file containing student marks or product sales), open it in Jupyter, clean it up, find some interesting facts about it, and create a few simple charts. This is a very useful skill that you can apply in projects, internships and jobs.

4.2 Importing Data from Different Sources

The very first step of any data analysis project is getting the data into our notebook. Data can come from many different places. As a BCA student, you may get data from a CSV file, an Excel sheet downloaded from your college portal, a JSON file from a web API, or a SQL database. Pandas has separate functions to read all these formats, and most of them work in a

similar way. The table below summarizes the most common data formats and the corresponding Pandas functions used to load them.

Format	Description	Common Use	Pandas Function
CSV	Plain text, values separated by commas	Most common format for sharing data	pd.read_csv()
Excel	Microsoft Excel files (.xlsx, .xls)	Business reports, accounting	pd.read_excel()
JSON	JavaScript Object Notation	APIs, web applications	pd.read_json()
SQL	Data stored in relational databases	Banks, hospitals, e-commerce	pd.read_sql()
API	Data fetched from web services	Twitter, weather, stock prices	requests.get() then read_json()
HTML	Tables in web pages	Reading tables from websites	pd.read_html()
Parquet	Compressed columnar format	Big data, data lakes	pd.read_parquet()

4.2.1 Reading CSV Files

CSV stands for Comma-Separated Values. It is the simplest and most common format for storing tabular data. A CSV file is just a text file in which each row contains values separated by commas. The first row usually contains the column names. CSV files can be opened in any text editor, in Microsoft Excel, or in any data analysis tool.

To read a CSV file in Pandas, we use the read_csv() function. Here is a simple example:

```
import pandas as pd

# Read the CSV file
df = pd.read_csv('students.csv')

# Display the first 5 rows
print(df.head())
```

In this code, df is short for DataFrame. It is a common convention among Pandas users to call the main DataFrame df. The head() method shows the first five rows of the DataFrame, which is useful to quickly check what the data looks like.

read_csv() has many useful options. Some commonly used ones are listed below.

1. sep: The character that separates the values. The default is a comma, but it can be a tab, semicolon or any other character.
2. header: The row number to use as column names. Default is 0 (the first row).
3. names: A list of column names to use, if the file does not have a header row.
4. index_col: The column to use as the row labels.
5. usecols: Read only specific columns instead of all.
6. nrows: Read only the first n rows. Useful for very large files.
7. skiprows: Skip the first n rows of the file.
8. encoding: The character encoding of the file (for example, 'utf-8' or 'latin-1').

4.2.2 Reading Excel Files

Excel files are very common in business. An Excel file (.xlsx or .xls) can have multiple sheets in a single workbook. Pandas can read Excel files using the read_excel() function. Note: you may need to install the openpyxl library first, using pip install openpyxl.

```
import pandas as pd

# Read a specific sheet
df = pd.read_excel('sales.xlsx', sheet_name='January')

# Read all sheets at once (returns a dictionary)
all_sheets = pd.read_excel('sales.xlsx', sheet_name=None)

# Access a specific sheet from the dictionary
jan_data = all_sheets['January']
```

4.2.3 Reading JSON Files

JSON stands for JavaScript Object Notation. It is a lightweight format that is easy for both humans and computers to read. JSON is the most common format used by web APIs (for example, the Twitter API, the weather API, and so on).

Pandas can read JSON files using read_json():

```
import pandas as pd

# Read a JSON file
df = pd.read_json('data.json')
print(df.head())
```

4.2.4 Reading from a SQL Database

Many real-world applications store their data in SQL databases such as MySQL, PostgreSQL, SQL Server or SQLite. Pandas can read data from any of these databases. We first connect to the database, write a SQL query and then use Pandas to load the result into a DataFrame.

Here is a small example using SQLite, which is a simple file-based database that comes with Python:

```
import pandas as pd
import sqlite3

# Connect to the SQLite database file
conn = sqlite3.connect('college.db')

# Write a SQL query
query = 'SELECT name, branch, marks FROM students WHERE marks > 80'

# Execute the query and load the result into a DataFrame
df = pd.read_sql(query, conn)

# Close the connection
conn.close()
print(df)
```

For other databases like MySQL or PostgreSQL, we can use the SQLAlchemy library, which provides a uniform way to connect to many different databases.

4.2.5 Fetching Data from an API

APIs (Application Programming Interfaces) allow us to fetch data from online services such as Twitter, Google, weather services and many more. To fetch data from an API, we use Python's

requests library. The data usually comes back in JSON format, which we can then convert to a DataFrame.

```
import pandas as pd
import requests

# Make a GET request to the API
url = 'https://api.example.com/users'
response = requests.get(url)

# Convert the JSON response to a Python dictionary
data = response.json()

# Convert to a DataFrame
df = pd.DataFrame(data)
print(df.head())
```

In real-world projects, APIs often require an authentication key. The exact way to use an API depends on its documentation, so always read the API documentation first.

4.3 Loading Data with Pandas and NumPy

We have already seen many examples of loading data with Pandas. Pandas is the main library used for loading and working with structured data in Python. NumPy is used more often for numerical arrays without column labels.

4.3.1 Pandas DataFrame

A Pandas DataFrame is a two-dimensional, labelled data structure. It can hold data of different types in different columns and provides hundreds of methods for working with that data. We can think of a DataFrame as something between a Python dictionary and an Excel sheet, but much more powerful than either.

There are several ways to create a DataFrame, including from a CSV file (as we saw above), from a Python dictionary, from a list of lists, from a NumPy array, or even from another DataFrame. Here is an example using a dictionary:

```
import pandas as pd

data = {
```

```
    'Roll_No': [101, 102, 103, 104, 105],
    'Name': ['Aarav', 'Diya', 'Karan', 'Mira', 'Rohan'],
    'Age': [19, 20, 19, 21, 20],
    'Marks': [78, 85, 90, 65, 92]
}

df = pd.DataFrame(data)
print(df)
```

4.3.2 NumPy Arrays

NumPy is mainly used when our data is purely numerical and does not have column labels. We typically use NumPy for mathematical computations, image data, and as input for machine learning algorithms.

```
import numpy as np

# Create a 1-D array
marks = np.array([78, 85, 90, 65, 92])

# Create a 2-D array (3 rows, 4 columns)
grid = np.array([
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10, 11, 12]
])

print('Shape:', grid.shape)  # (3, 4)
print('Mean:', np.mean(grid))
```

4.3.3 Choosing Between Pandas and NumPy

As a general rule, use Pandas when your data has columns with names (like a CSV file), and use NumPy when your data is just numbers (like images or matrices). In practice, Pandas DataFrames are built on top of NumPy arrays, so the two libraries work very well together.

4.4 Importing Data from Local and Remote Sources

Data can be stored either on our own computer (local) or on the internet (remote). Pandas can read from both, with very small changes.

4.4.1 Local Sources

Local sources are files on our own computer. We just give the path to the file:

```
# Using a relative path (in the current folder)
df = pd.read_csv('students.csv')


# Using an absolute path
df = pd.read_csv('C:/Users/Sanskruti/Documents/students.csv')
```

On Windows, paths normally use backslashes (\). However, in Python, backslash is used as an escape character, so we either use forward slashes (/) or double backslashes (\\).

4.4.2 Remote Sources

Remote sources are files hosted on the internet. We can give the URL of the file directly to Pandas, and it will download and read the file for us:

```
# Read a CSV directly from a URL
url = 'https://example.com/data/students.csv'
df = pd.read_csv(url)
print(df.head())
```

If we want to download the file first and save it locally, we can use the requests library:

```
import requests
import pandas as pd

url = 'https://example.com/data/students.csv'
response = requests.get(url)

# Save the file locally
with open('students.csv', 'wb') as f:
    f.write(response.content)

# Now read the local file
df = pd.read_csv('students.csv')
```

4.5 Handling Large Datasets: Chunking and Sampling

Modern datasets can be very large — sometimes containing millions of rows and many columns. Loading such datasets entirely into the computer's memory can be slow or even impossible. Pandas provides two simple techniques to handle this problem: chunking and sampling.

4.5.1 Chunking

Chunking means reading the file in small pieces (chunks) instead of all at once. We can specify the chunk size and process each chunk one by one. This way, only one chunk is in memory at any time, no matter how large the file is.

```
import pandas as pd

chunk_size = 10000  # 10,000 rows per chunk
chunks = pd.read_csv('big_file.csv', chunksize=chunk_size)

total = 0
count = 0

for chunk in chunks:
    # Process each chunk
    total += chunk['marks'].sum()
    count += len(chunk)

average = total / count
print('Average marks:', average)
```

In this code, we calculate the average marks across a very large file by reading 10,000 rows at a time and keeping a running total. Notice that we never load the full file into memory.

4.5.2 Sampling

Sampling means selecting a small random subset of the data. This is useful for exploration when we do not need all the data; we just want to get a feel for what it looks like. Pandas provides the sample() method for this:

```
import pandas as pd

df = pd.read_csv('big_file.csv')
```

```
# Pick 1000 random rows
sample_df = df.sample(n=1000)

# Or pick 5% of the rows
sample_df = df.sample(frac=0.05)

# random_state makes the sample reproducible
sample_df = df.sample(n=1000, random_state=42)
```

Sampling is also used in machine learning to split data into training and testing sets. We will see this in later courses.

Check Your Progress - 1

1. Which Pandas function is used to read each of the following file formats: (a) CSV, (b) Excel, (c) JSON, (d) SQL?
2. What is the difference between a local source and a remote source?
3. True or False: chunksize in pd.read_csv() loads the entire file into memory at once.
4. What is the purpose of sampling? Give one situation where sampling is useful.
5. Write a single line of Python code to read a CSV file from the URL https://example.com/data.csv into a DataFrame called df.
6. Why is the parameter random_state used in df.sample()?

4.6 Data Exploration: Getting to Know Your Data

Once we have loaded our data into a DataFrame, the next step is to explore it. Data exploration helps us understand the structure, size, content and quality of our data. Without this step, we may make wrong assumptions and produce misleading results.

Data exploration usually involves answering questions like: How many rows and columns does the data have? What types of values are in each column? Are there any missing values? Are there any obvious errors? What is the average, minimum and maximum of the numeric columns? What are the most common values in the categorical columns?

Pandas provides several easy methods to answer these questions. The most useful ones are listed below, with examples.

4.6.1 First Look at the Data

After loading a DataFrame, the first thing to do is look at the first few rows. The head() and tail() methods help with this.

```
df.head()      # First 5 rows (default)
df.head(10)    # First 10 rows
df.tail()      # Last 5 rows
df.tail(3)     # Last 3 rows
```

If we want to see a few random rows from the middle, we can use sample():

```
df.sample(5)   # 5 random rows
```

4.6.2 Shape and Size

To find the size of the DataFrame, use shape, size and len():

```
df.shape       # (rows, columns) - returns a tuple
df.size        # Total number of cells = rows * columns
len(df)        # Number of rows
len(df.columns)  # Number of columns
```

4.6.3 Column Names and Data Types

To know what columns the DataFrame has, use columns. To know the data type of each column, use dtypes:

```
df.columns      # Names of all columns
df.dtypes       # Data type of each column
df.info()       # A complete summary
```

info() is one of the most useful methods. It shows column names, data types, the number of non-missing values in each column, and the memory usage of the DataFrame.

4.6.4 Selecting Columns and Rows

We often need to look at specific columns or specific rows. Pandas provides several easy ways to do this.

```
# Select a single column
df['Marks']


# Select multiple columns
df[['Name', 'Marks']]
```

```
# Select rows where a condition is true
df[df['Marks'] > 80]

# Multiple conditions (use & for AND, | for OR)
df[(df['Marks'] > 80) & (df['Branch'] == 'BCA')]

# Select rows by position
df.iloc[0]      # First row
df.iloc[0:5]    # First 5 rows

# Select rows by label
df.loc[101]     # Row with index label 101
```

4.7 Summary Statistics and Data Types

After getting a basic feel for the data, we usually want to look at summary statistics. Summary statistics give us a quick numerical overview of the data.

4.7.1 Common Summary Statistics

Some common summary statistics are listed below.

1. Mean: The average of all values.
2. Median: The middle value when data is arranged in order.
3. Mode: The most frequent value.
4. Standard Deviation: A measure of how spread out the values are around the mean.
5. Minimum and Maximum: The smallest and largest values.
6. Quantiles or Percentiles: Values that divide the data into equal parts. The 25th, 50th and 75th percentiles are commonly used.
7. Count: The number of non-missing values.

4.7.2 Calculating Summary Statistics in Pandas

Pandas makes it very easy to calculate these statistics. The describe() method gives all of them at once for the numeric columns:

```
df.describe()
```

If we want a single statistic, we can use the corresponding method:

```
df['Marks'].mean()      # Average
```

```
df['Marks'].median()    # Middle value
df['Marks'].std()       # Standard deviation
df['Marks'].min()       # Minimum
df['Marks'].max()       # Maximum
df['Marks'].count()     # Count of non-missing values
df['Marks'].sum()       # Total
df['Branch'].mode()     # Most frequent value
df['Branch'].value_counts()  # Frequency of each value
```

4.7.3 Understanding Data Types

Each column in a DataFrame has a specific data type. Knowing the data type is important because different operations work on different types. The most common data types are listed below.

1. int64: Integer numbers (whole numbers without decimals).
2. float64: Floating-point numbers (numbers with decimals).
3. object: Text or mixed data (strings).
4. bool: Boolean (True or False).
5. datetime64: Dates and times.
6. category: Categorical data (a fixed set of categories like 'Male'/'Female').

To check data types, use dtypes. To change the data type of a column, use astype():

```
# Check data types
df.dtypes


# Convert a column to a different type
df['Age'] = df['Age'].astype(int)
df['Marks'] = df['Marks'].astype(float)
df['Branch'] = df['Branch'].astype('category')
```

Sometimes a column that looks like a number is actually stored as text (object type). This often happens with CSV files. We must convert it to a numeric type before we can do calculations on it.

4.8 Handling Missing Data and Duplicates

Real-world data is rarely perfect. It often has missing values (empty cells) and duplicate rows. Both of these problems must be addressed before any serious analysis.

4.8.1 Identifying Missing Data

In Pandas, missing values are usually represented by NaN (Not a Number). We can find missing values using isnull() or isna() (these are the same).

```
# Check which cells are missing
df.isnull()

# Count missing values in each column
df.isnull().sum()

# Total number of missing values in the entire DataFrame
df.isnull().sum().sum()

# Percentage of missing values in each column
(df.isnull().sum() / len(df)) * 100
```

4.8.2 Handling Missing Data

There are two main ways to handle missing data: drop the rows that contain missing values, or fill the missing values with some sensible value.

To drop rows with missing values:

```
# Drop all rows that have at least one missing value
df_clean = df.dropna()

# Drop rows where the 'Marks' column is missing
df_clean = df.dropna(subset=['Marks'])

# Drop columns that have any missing values
df_clean = df.dropna(axis=1)
```

To fill missing values:

```
# Fill all missing values with 0
df_filled = df.fillna(0)

# Fill missing values in 'Marks' with the mean
mean_marks = df['Marks'].mean()
```

```
df['Marks'] = df['Marks'].fillna(mean_marks)

# Fill missing values in 'Branch' with the most frequent value
df['Branch'] = df['Branch'].fillna(df['Branch'].mode()[0])

# Forward fill (use the previous value)
df = df.fillna(method='ffill')
```

Choosing the right method depends on the situation. If only a few rows have missing values, it is often safe to drop them. If many rows have missing values in one column, dropping the column may be better. If we want to keep all rows, filling with the mean (for numbers) or mode (for text) is a common approach.

4.8.3 Handling Duplicates

Duplicate rows can also cause problems in our analysis. Pandas provides easy methods to find and remove them.

```
# Check which rows are duplicates
df.duplicated()

# Count the number of duplicate rows
df.duplicated().sum()

# Remove duplicate rows, keeping the first occurrence
df_unique = df.drop_duplicates()

# Remove duplicates based on specific columns only
df_unique = df.drop_duplicates(subset=['Roll_No'])

# Keep the last occurrence instead of the first
df_unique = df.drop_duplicates(keep='last')
```

.

4.8 Data Transformation and Formatting for Analysis

Data transformation is the process of converting data into the correct format or structure for analysis. Some common transformations include:

- Type Conversion: Changing the data type of columns, such as converting strings to numerical values (df['column'].astype(int)).
- Normalization/Standardization: Scaling data to a fixed range or mean (e.g., using Min-Max scaling or Z-score normalization) to bring different features to comparable scales.
- Feature Engineering: Creating new features from existing data, such as extracting year or month from a date field or creating binary columns (e.g., "is_active") from categorical data.
- Merging and Aggregating Data: Combining multiple datasets using operations like merge() or concat() and grouping data for aggregation (e.g., groupby()).
- These transformations are necessary to ensure that the data is ready for statistical analysis or machine learning models.

4.9 Data Transformation and Formatting

Data transformation means converting the data into the right format for our analysis or model. Some common transformations are explained below.

4.9.1 Type Conversion

As we saw earlier, sometimes a column is stored in the wrong data type. We use astype() to fix this:

```
df['Age'] = df['Age'].astype(int)
df['Marks'] = df['Marks'].astype(float)
df['Date'] = pd.to_datetime(df['Date'])
```

4.9.2 Renaming Columns

Column names sometimes need to be cleaned up, for example to remove spaces or to make them more readable:

```
# Rename specific columns
df = df.rename(columns={
    'std_marks': 'Student_Marks',
    'br': 'Branch'
})

# Convert all column names to lowercase
df.columns = df.columns.str.lower()
```

4.9.3 Normalization and Standardization

When the columns have very different ranges, machine learning models can become biased. For example, age may be from 18 to 25, while income may be from 10,000 to 100,000. Normalization and standardization put all columns on a similar scale.

Normalization scales the data to a fixed range, usually 0 to 1, using the formula:

```
X_norm = (X - X_min) / (X_max - X_min)
```

Standardization changes the data so that it has a mean of 0 and a standard deviation of 1:

```
X_std = (X - mean) / std
```

In Python, we can do this manually or use Scikit-learn:

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler

# Normalization (0 to 1)
scaler = MinMaxScaler()
df['Marks_norm'] = scaler.fit_transform(df[['Marks']])

# Standardization (mean 0, std 1)
scaler = StandardScaler()
df['Marks_std'] = scaler.fit_transform(df[['Marks']])
```

4.9.4 Feature Engineering

Feature engineering means creating new columns from existing data. New features often help in analysis and modelling. A few examples are listed below.

```
# Extract year and month from a date
df['Year'] = df['Date'].dt.year
df['Month'] = df['Date'].dt.month

# Create a binary column from a numeric one
df['Pass'] = (df['Marks'] >= 40).astype(int)

# Create a category from a numeric column
def grade(marks):
    if marks >= 90: return 'A+'
    elif marks >= 75: return 'A'
```

```
    elif marks >= 60: return 'B'
    elif marks >= 40: return 'C'
    else: return 'F'

df['Grade'] = df['Marks'].apply(grade)
```

4.9.5 Merging and Aggregating Data

Often we have data spread across multiple DataFrames that we need to combine. We can merge two DataFrames on a common column:

```
students = pd.DataFrame({
    'Roll_No': [101, 102, 103],
    'Name': ['Aarav', 'Diya', 'Karan']
})

marks = pd.DataFrame({
    'Roll_No': [101, 102, 103],
    'Marks': [78, 85, 90]
})

# Combine the two on Roll_No
combined = pd.merge(students, marks, on='Roll_No')
print(combined)
```

To aggregate (group and summarize) data, use groupby():

```
# Average marks by branch
df.groupby('Branch')['Marks'].mean()

# Multiple statistics at once
df.groupby('Branch')['Marks'].agg(['mean', 'min', 'max', 'count'])
```

4.10 Data Visualization Basics and EDA

Data visualization is a very important part of exploration. As the saying goes, a picture is worth a thousand words. Charts often reveal patterns that we cannot see by just looking at numbers.

4.10.1 Quick Plots with Pandas

Pandas has a built-in plot() method that uses Matplotlib in the background. This makes simple plots very quick:

```
import matplotlib.pyplot as plt

# Histogram of marks
df['Marks'].hist(bins=10)
plt.title('Distribution of Marks')
plt.xlabel('Marks')
plt.ylabel('Number of Students')
plt.show()

# Bar chart of branches
df['Branch'].value_counts().plot(kind='bar')
plt.title('Number of Students per Branch')
plt.show()

# Scatter plot of two columns
df.plot(kind='scatter', x='Age', y='Marks')
plt.show()

# Box plot to see outliers
df.boxplot(column='Marks', by='Branch')
plt.show()
```

4.10.2 Common Plots for EDA

Different plots are used to answer different questions. The table below summarizes the most common plots and what they show.

1. Histogram: Shows the distribution of a numeric column. Useful for seeing if the data is normal, skewed or has gaps.
2. Box Plot: Shows the median, quartiles and outliers of a numeric column. Useful for spotting unusual values.

3. Bar Chart: Shows the count or sum of a value for each category. Good for comparing groups.
4. Pie Chart: Shows proportions. Use sparingly because pie charts are often hard to read.
5. Scatter Plot: Shows the relationship between two numeric columns. Each point is one row of data.
6. Line Plot: Shows how a value changes over time. Best when the x-axis is time.
7. Heatmap: Shows the values of a 2-D matrix using colour. Often used to show correlations.
8. Pair Plot: Shows scatter plots between every pair of numeric columns. Useful at the start of an analysis.

4.10.3 EDA with Seaborn

Seaborn makes statistical plots especially easy and attractive:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Histogram with KDE (smooth curve)
sns.histplot(df['Marks'], kde=True)
plt.show()

# Box plot grouped by category
sns.boxplot(data=df, x='Branch', y='Marks')
plt.show()

# Scatter plot with regression line
sns.regplot(data=df, x='Age', y='Marks')
plt.show()

# Heatmap of correlations
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.show()

# Pair plot of all numeric columns
sns.pairplot(df)
```

```
plt.show()
```

4.11 A Complete Mini Example

Let us put everything together with a small but complete example. Suppose we have a CSV file students.csv that contains the following columns: Roll_No, Name, Branch, Age, Marks. The file may have some missing values and a few duplicate rows. We want to load it, clean it, explore it and create a few charts. Here is the complete code:

```
# Step 1: Import the libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Step 2: Load the data
df = pd.read_csv('students.csv')
print('Shape:', df.shape)
print(df.head())

# Step 3: First look
print(df.info())
print(df.describe())

# Step 4: Check missing values
print('Missing values:')
print(df.isnull().sum())

# Step 5: Handle missing values
# Fill numeric columns with the mean
df['Marks'] = df['Marks'].fillna(df['Marks'].mean())
df['Age'] = df['Age'].fillna(df['Age'].median())

# Fill text columns with the most frequent value
df['Branch'] = df['Branch'].fillna(df['Branch'].mode()[0])
```

```
# Step 6: Remove duplicates
before = len(df)
df = df.drop_duplicates()
print(f'Removed {before - len(df)} duplicate rows')

# Step 7: Create a new column
df['Pass'] = (df['Marks'] >= 40).astype(int)

# Step 8: Group by branch
branch_summary = df.groupby('Branch').agg(
    avg_marks=('Marks', 'mean'),
    pass_rate=('Pass', 'mean'),
    count=('Roll_No', 'count')
)
print(branch_summary)

# Step 9: Visualize
# Histogram of marks
plt.figure(figsize=(8, 5))
sns.histplot(df['Marks'], kde=True, bins=15)
plt.title('Distribution of Marks')
plt.show()

# Box plot by branch
plt.figure(figsize=(8, 5))
sns.boxplot(data=df, x='Branch', y='Marks')
plt.title('Marks by Branch')
plt.show()

# Save the cleaned data
df.to_csv('students_cleaned.csv', index=False)
print('Done!')
```

This example shows the typical flow of a data analysis project: load, explore, clean, transform, visualize and save. As you become more comfortable, you will add more steps and customize each step to your needs. But the basic flow remains the same.

Check Your Progress - 2

1. Which Pandas method gives a quick statistical summary of all numeric columns at once?
2. Fill in the blank: To find duplicate rows in a DataFrame, we use df._____().
3. Name two ways to handle missing values in a column.
4. What is the difference between normalization and standardization?
5. Which type of plot is best for: (a) seeing the distribution of marks, (b) finding outliers, (c) comparing the count of students in each branch?
6. True or False: Visualization is only useful for presenting final results, not during exploration.

4.13 Let Us Sum Up

In this unit, we have studied the practical steps of working with data in Jupyter Notebooks. We started by learning how to import data from many different sources, including CSV files, Excel files, JSON files, SQL databases and web APIs. We saw that Pandas has a single, easy function for each of these formats.

We then learned how to load data using Pandas and NumPy, and how to read data from both local files and remote URLs. Next, we explored the data using Pandas methods such as head(), tail(), shape, info() and describe(). We learned how to select specific columns and rows. We then studied the most common summary statistics (mean, median, mode, standard deviation, min, max, quantiles) and how to calculate each of them in Pandas. We also learned about the most common data types and how to convert between them.

We covered handling of missing values using isnull(), dropna() and fillna(), and handling of duplicates using duplicated() and drop_duplicates(). Finally, we looked at simple visualizations for exploratory data analysis using Pandas' built-in plotting and Seaborn. We discussed common chart types and when to use each. The unit ended with a complete mini example that brought all these steps together. By the end of this unit, you should be able to take a small dataset and perform a complete exploratory analysis on it.

4.14 Check your progress: Possible Answers

Pandas functions: (a) pd.read_csv() for CSV, (b) pd.read_excel() for Excel, (c) pd.read_json() for JSON, (d) pd.read_sql() for SQL.

A local source is a file stored on our own computer that we read using a file path (for example, 'students.csv'). A remote source is a file stored somewhere on the internet that we read using a URL (for example, 'https://example.com/data.csv'). Pandas can read from both, with little change in the code.

False. The chunksize parameter in pd.read_csv() loads the file in small pieces (chunks) instead of loading the entire file into memory at once. This is useful for very large files.

Sampling is the process of selecting a small random subset of the data. It is useful when the dataset is too large to work with comfortably, or when we just want to quickly explore the data without processing all of it.

df = pd.read_csv('https://example.com/data.csv')

The random_state parameter in df.sample() makes the sample reproducible. Whenever we run the code with the same random_state value, we get the same random sample. This is useful for debugging and for ensuring consistent results across runs.

The Pandas method df.describe() gives a quick statistical summary of all numeric columns at once. It shows count, mean, standard deviation, min, max, and the 25th, 50th and 75th percentiles.

df.duplicated()

Two ways to handle missing values: (a) drop rows or columns containing missing values using df.dropna(), or (b) fill missing values with a specific value, such as the mean, median, mode, or a constant, using df.fillna().

Normalization scales the data to a fixed range, usually [0, 1], using the formula (X - X_min) / (X_max - X_min). Standardization changes the data so that it has a mean of 0 and a standard deviation of 1, using the formula (X - mean) / std. Normalization is useful when we want a bounded range, while standardization is preferred when the data is approximately normally distributed.

Plot choice: (a) histogram for the distribution of marks, (b) box plot for finding outliers, (c) bar chart for comparing the count of students in each branch.

False. Visualization is very useful during exploration, not only for presenting final results. Charts during exploration help us spot patterns, outliers and relationships that are hard to see in raw numbers.

4.15 Further Readings

1. Wes McKinney, "Python for Data Analysis", O'Reilly Media, 3rd Edition, 2022.
2. Marc Wintjen and Andrew Vlahutin, "Practical Data Analysis Using Jupyter Notebook", Packt Publishing, 2020.
3. Jake VanderPlas, "Python Data Science Handbook", O'Reilly Media, 2nd Edition, 2023.
4. Online resource: Pandas tutorial at https://pandas.pydata.org/docs/getting_started/intro_tutorials/
5. Online resource: Kaggle Learn Pandas course at https://www.kaggle.com/learn/pandas
6. Online resource: Real Python tutorial on data cleaning at https://realpython.com/python-data-cleaning-numpy-pandas/

4.16 Assignments

1. Explain the different sources of data we can import into a Jupyter Notebook. Give the Pandas function used to read each format.
2. Write a Python program that loads a CSV file, displays the first 10 rows, prints the shape and the data types of each column, and shows the summary statistics.
3. Explain chunking and sampling. When would you prefer one over the other?
4. What is missing data? Describe at least three ways to handle missing values in a Pandas DataFrame, with an example of each.
5. Explain the difference between dropna() and fillna() with examples. When should we use each?
6. Match each plot with its best use case: (a) Histogram, (b) Box plot, (c) Scatter plot, (d) Bar chart, (e) Heatmap. Use cases: (i) finding outliers, (ii) showing distribution of one variable, (iii) showing relationship between two numeric variables, (iv) showing correlation between many variables, (v) comparing counts across categories.
7. Practical Assignment: Download any small public dataset (for example, the Iris dataset, the Titanic dataset or the marks of students in your class). Load it into a Jupyter Notebook, clean any missing values and duplicates, perform exploratory analysis with summary statistics and at least three different charts, and write a short report (about 200 words) describing the insights you found.

BAOU
Education for All

યુનિવર્સિટી ગીત

સ્વાધ્યાયઃ પરમં તપઃ
સ્વાધ્યાયઃ પરમં તપઃ
સ્વાધ્યાયઃ પરમં તપઃ

શિક્ષણ, સંસ્કૃતિ, સદ્‌ભાવ, દિવ્યબોધનું ધામ
ડૉ. બાબાસાહેબ આંબેડકર ઓપન યુનિવર્સિટી નામ;
સૌને સૌની પાંખ મળે, ને સૌને સૌનું આભ,
દશે દિશામાં સ્મિત વહે હો દશે દિશે શુભ-લાભ.

અભણ રહી અજ્ઞાનના શાને, અંધકારને પીવો ?
કહે બુદ્ધ આંબેડકર કહે, તું થા તારો દીવો;
શારદીય અજવાળા પહોંચ્યાં ગુર્જર ગામે ગામ
ધ્રુવ તારકની જેમ ઝળહળે એકલવ્યની શાન.

સરસ્વતીના મયૂર તમારે ફળિયે આવી ગહેકે
અંધકારને હડસેલીને ઉજાસના ફૂલ મહેંકે;
બંધન નહીં કો સ્થાન સમયના જવું ન ઘરથી દૂર
ઘર આવી મા હરે શારદા દૈન્ય તિમિરના પૂર.

સંસ્કારોની સુગંધ મહેંકે, મન મંદિરને ધામે
સુખની ટપાલ પહોંચે સૌને પોતાને સરનામે;
સમાજ કેરે દરિયે હાંકી શિક્ષણ કેરું વહાણ,
આવો કરીયે આપણ સૌ
ભવ્ય રાષ્ટ્ર નિર્માણ...
દિવ્ય રાષ્ટ્ર નિર્માણ...
ભવ્ય રાષ્ટ્ર નિર્માણ

DR. BABASAHEB AMBEDKAR OPEN UNIVERSITY
(Established by Government of Gujarat)
'Jyotirmay' Parisar,
Sarkhej-Gandhinagar Highway, Chharodi, Ahmedabad-382 481
Website : www.baou.edu.in