

Data Science Toolbox

DECAP792

Edited by:
Ajay Kumar Bansal



L OVELY
P ROFESSIONAL
U NIVERSITY



Data Science Toolbox

**Edited By
Ajay Kumar Bansal**

Title: DATA SCIENCE TOOLBOX

Author's Name: Dr. Divya

Published By : Lovely Professional University

Publisher Address: Lovely Professional University, Jalandhar Delhi GT road, Phagwara - 144411

Printer Detail: Lovely Professional University

Edition Detail: (I)

ISBN: 978-81-19334-46-9



Copyrights@ Lovely Professional University

Content

Unit 1:	Introduction to Data Science	1
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 2:	Data Pre-Processing	11
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 3:	Various Data Pre-processing Operations	17
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 4:	Data Plotting and Visualization	30
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 5:	Role of Statistics in Data Science	54
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 6:	Machine Learning	68
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 7:	Unsupervised Learning	77
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 8:	Supervised Learning	92
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 9:	Regression Models	105
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 10:	Weka	115
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 11:	Excel Data Analysis	130
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 12:	R Tool	142
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 13:	R Tool	171
	<i>Dr. Divya, Lovely Professional University</i>	
Unit 14:	NumPy and Pandas	200
	<i>Dr. Divya, Lovely Professional University</i>	

Unit 01: Introduction to Data Science

CONTENTS

Objectives

Introduction

1.1 Data Classification

1.2 Data Collection

1.3 Why Learn Data Science?

1.4 Data Analytic Lifecycle

1.5 Types of Data Analysis

1.6 Some of the Key Stakeholders

1.7 Types of Jobs in Data Analytics

1.8 Pros and Cons of Data Science

Summary

Keywords

Self Assessment

Answer for Self Assessment

Review Questions

Further Readings

Objectives:

After studying this unit, you will be able to

- understand the concept of data science,
- understand the need of data science,
- understand the life cycle of data analytic,
- understand the types of data analytic,
- understand the pros and cons of data science.

Introduction

Data science is the task of scrutinizing and processing raw data to reach a meaningful conclusion. A lot of data is mined and classified to detect and study the behavioral data and patterns. Every year the data is growing exponentially in zettabytes which requires processing. The below image shows the tentative increase in data in coming years. This image shows the demand of data science tools for the processing of data so that meaningful conclusions can be drawn. There are a number of tools available for processing the raw data like Weka, RapidMiner, R tool, Excel, Python, Tableau, KNIME, PowerBI and DataRobot etc.

1.1 Data Classification

The data available is nominal, ordinal, interval, and ratio data.

Nominal Data: It contains a set of items that can be distinguished by the name of category.

Ordinal Data: It contains the items that can be ordered, such as military rank, or units of government, but whose degree of difference can't be measured.

Data Science Toolbox

Interval Data: It contains the items that have a measurable distance between them, but no meaningful zero point, such as Fahrenheit and Celsius temperatures.

Ratio Data: The measurements that have a meaningful zero and can be divided meaningfully, such as the Kelvin temperature scale.

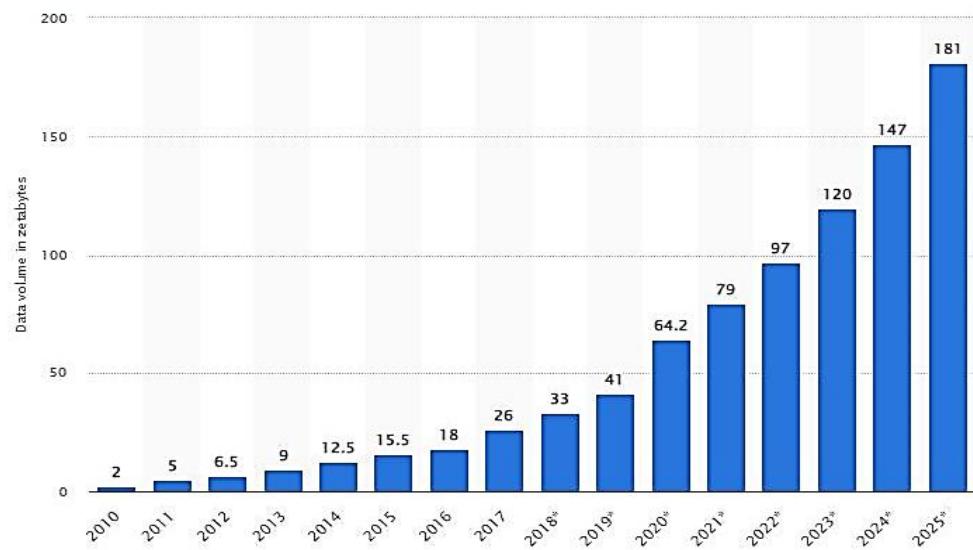


Figure 1: Increase in data

1.2 Data Collection

Two major sources of data are:

Primary data:

A primary data source is an original data source, that is, one in which the data are collected first-hand by the researcher for a specific research purpose or project. It can be collected in a number of ways. The most common techniques are surveys, interviews, observations and experiments. The primary data collection is quite expensive and time-consuming as compared to secondary data collection.

Secondary data:

The secondary data is the data that has been already collected through primary sources and made readily available for researchers to use for their own research. It is a type of data that has already been collected in the past. Sources of secondary data include books, personal sources, journals, newspapers, websites, government records etc. Secondary data are known to be readily available compared to that of primary data. It requires very little research and needs for manpower to use these sources.

The popular websites for data downloading are UCI machine learning repository, Kaggle datasets, IMDB datasets, Stanford Large Network Dataset Collections.

1.3 Why Learn Data Science?

The data scientists use the tools of data science in various sectors like ecommerce, finance, retail, healthcare, education, human resource, sports and various others. The use in every field is described below.

- **Ecommerce:** In ecommerce, the tools of data science are used for maximizing revenue and profitability.
- **Finance:** In finance, the different tools are available to take care of risk analysis, fraud detection, working capital management.

- **Retail:** For optimal pricing, better marketing strategies, stock management, the data science toolbox can be used.
- **Healthcare:** For patient quality care, classification of the type of symptoms of patients and predicted health deficiencies, we can use data science tools.
- **Education:** In academic institutions the data science toolbox can be deployed for better admission scenario, empowerment of students for successful examination results and all-round student performance.
- **Human resource:** For building strong leadership, employee acquisition, employee retention and performance management, we can use data science toolbox.
- **Sports:** To analyse the performance of players, the predicted scores, prevention of injuries and the possibility of winning or losing a match by a particular team. In sports, the data science tools can be used for all these tasks.

1.4 Data Analytic Lifecycle

Data science is an umbrella term, and the data analytics is a subset of data science. Data analytics is comprised of six phases that are carried out in a cycle. All these phases of data analytic lifecycle proceed one by one in line.

- 1) Data discovery
- 2) Data preparation
- 3) Planning of data models
- 4) Building of data models
- 5) Communication of results
- 6) Operationalization

Data Discovery

The stakeholders regularly perform the following tasks: examine the business trends, make case studies of similar data analytics, and study the domain of business industry. The team assesses in-house resources, in-house infrastructure, technology requirements and total time involved. After evaluations and assessments, stakeholders start formulating the initial hypotheses for resolving all business challenges in terms of current market scenario.

Data Preparation

Data is prepared by transforming it from a legacy system into a data analytics form by using any kind of platform. Example: IBM Netezza 1000 is one such Sandbox platform used by IBM Company for handling data marts.

Model Planning

Proper planning of methods to be adopted and the various workflows to be followed during the next phase is done in model planning phase. The various division of work among the team is decided. Feature selection is performed for applying it to the model.

Model Building

Building a model in which team works for deploying datasets for training and testing as well as for production purposes. The model is executed.



Figure 2: Data Analytic Lifecycle

Communicate Results

Checks the results of project to find whether it is a success or failure. The inferences are drawn on key insights, the entire work is summarized and the elaborate narrative on the key findings is prepared.

Operationalization

A final report is made by the team along with the briefings, source codes and related documents. It involves running the pilot project in real time and test the project in real time environment.

1.5 Types of Data Analysis

Descriptive analysis

It is the simplest and most common type of data analysis. The main emphasis is on “What has happened?” by analysing valuable information found from the available past data. The data dealt with are large in volume and often includes the entire population. Mostly used in businesses to generate monthly revenue reports, sales leads and KPI indicators. For example: A data analyst will be able to generate the statistical results of the performance of the cricket players of team India.

Diagnostic analysis

The emphases are not only on ‘what has happened?’ but also on ‘why it happened?’. Deeper insights on reasons behind the pattern of data found in the past. For example: A data analyst will be able to find why the performance of each cricket player of India has risen/degraded in recent past six months. The major task for the analyst must be correct enough to find the reason behind a particular change or cause of occurrence to make a gain or profit in various fields. Machine learning techniques are used very often for this.

Predictive analysis

It deals with the prediction of future based upon the available current and past data. The emphasis is given on ‘what is likely to happen?’. For example: A data analyst will be able to predict the performance of each player of Indian cricket team. Predictive analysis is applied in many domains such as risk management, weather forecasting, sales forecasting and prediction of the performance of each team. Usually, prediction is made by dividing the available dataset in training and testing sets. The predicted solution provides an approximated forecast results and may vary from the actual results.

Prescriptive analysis

It is highest in terms of complexity. The insights gain from other three types of analysis are combined to determine the kind of action to be taken to solve a certain situation. It involves a high degree of responsibility, time and complicity to reach to informed decision making. It can be summarised as

- 1) Descriptive analysis – what has happened till date.
- 2) Diagnostic analysis – Why it has happened in a particular way.
- 3) Predictive analysis – what might happen in near future.
- 4) Prescriptive analysis – recommendations based on forecast.

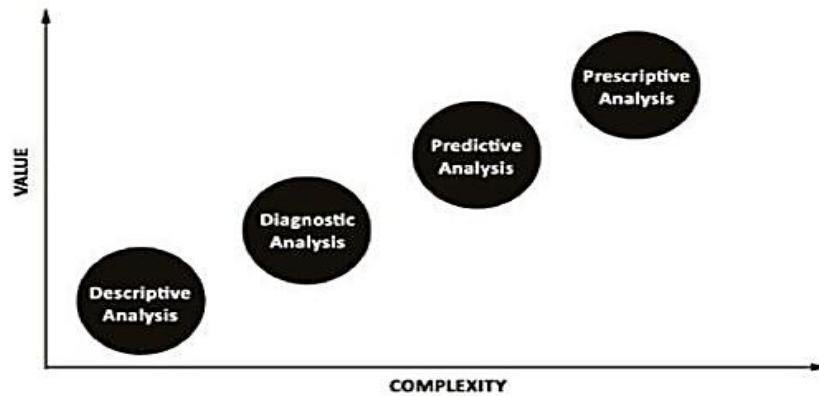


Figure 3: Data Analytic Types

1.6 Some of the Key Stakeholders

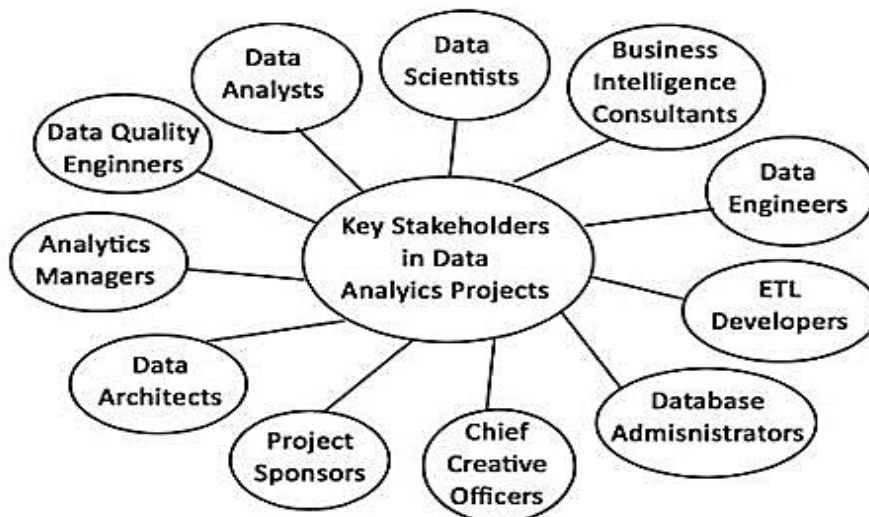


Figure 4: Stakeholders

Each of the stakeholder has a clear role to play for a business problem

- understanding the essentials of the problem,
- proper planning,
- implementation of the project,
- analysing various outcomes of the project,
- solving bottlenecks visible in the outcomes,
- generating reports by drawing inferences about the success of the project.

1.7 Types of Jobs in Data Analytics

Various stakeholders in data analytics are:

- 1) Data analyst

Data Science Toolbox

- 2) Data scientist
- 3) Data engineer
- 4) Database administrator
- 5) Analytics manager

Data analyst

The main role is to extract data and interpret the information attained from the data for analysing the outcome of a given problem in business. The analyst also discovers the various bottlenecks that are found in the results and provides possible solutions for the same. Information is extracted from given existing data is done using one or more standard methodologies such as data cleaning, data transformation, data visualization and data modelling. Major skills required to be data analyst:

- 1) Python
- 2) R programming
- 3) Structured Query Language
- 4) Statistical Analysis Software
- 5) SAS Miner
- 6) Microsoft Excel
- 7) Tableau.

Key areas and techniques for data analyst:

- 1) Data pre-processing: It involves data cleaning, data integration, data transformation and data reduction.
- 2) Data visualization: It involves graphical representation of data for easier analysis and better understanding.
- 3) Statistical modelling: It involves descriptive statistics and inferential statistics.
- 4) Programming skills: R and/or python programming.
- 5) Communication and presentation skills: Required for communicating with the team

Data Scientist

A data scientist incurs all the skills of a data analyst with the additional skills of data wrangling, complex machine learning, Big Data tools and software engineering. Both data analysts and data scientists use same tools and practices. However, the scope and nature of the problem discussed are different. The scope and nature of the problem addressed by a data scientist differ from data analyst. Data scientist deal with large and complex data that can be of high dimension, and carry out appropriate machine learning and visualization tools to convert the complex data into easily interpretable meaningful information. The fundamentals prerequisites for a data scientist

- 1) Statistics - Descriptive statistics and inferential statistics.
- 2) Mathematics - Linear algebra and calculus.
- 3) Computer programming - Python or R.
- 4) Database handling - SQL.

Data Engineer

The role of data engineer is not to analyse data but rather to prepare, manage and convert data into a form that can be readily used by a data scientist or data analyst. So, the job of data engineer comes first and then the data is handed over to data analyst or data scientist for analysis. With special training a data engineer can design, build, integrate and maintain data from multiple sources. The prominent work of data engineer:

- 1) Developing and maintaining data architectures.
- 2) Aligning data architectures with business or project requirements.
- 3) Improving data quality and raising data efficiency.
- 4) Performing predictive and prescriptive modelling for given input data.

- 5) Determining activities that can be automated.

Database Administrator

DBA administers and operates the database. The technical skills required by DBA are:

- 1) SQL
- 2) Scripting
- 3) Database performance tuning
- 4) System and network design
- 5) Backup and recovery of databases

Few prominent works done by DBA are:

- 1) Database designing as per end user requirements.
- 2) Providing/or revoking rights to/or from database end users.
- 3) Enable efficient data backup and data recovery mechanisms.
- 4) Database related training to end users.
- 5) Ensuring data privacy and security.
- 6) Managing data integrity for end users.
- 7) Monitoring the performances of the database.

Data Architect

Data architect provides the support of various tools and platforms that are required by data engineers to carry out various tests with precision. The skills required by data architects:

- 1) Data modelling
- 2) Data warehousing
- 3) Extraction, transformation and load (ETL)
- 4) Knowledge of Hive, Pig and Spark.

The prominent works of data architect are:

- 1) Design and implement database systems
- 2) Design and implement data models
- 3) Design and implement components of data architecture
- 4) Management of data warehouses and ETL operations
- 5) Knowledge of data available

Analytic Manager

The analytic manager is involved in overall management of various data analytics operations. The analytics manager deals with the team leader of each group and monitors and manages the work of each team. The major skills required to be an analytics manager are:

- 1) Python/R programming language
- 2) SQL
- 3) SAS

The prominent works of analytics manager are:

- 1) Leading data analysts' team
- 2) Having a thorough understanding of business requirements and objectives
- 3) Configuring and implementing data analytics solution
- 4) Ensuring the quality results of the reports developed by every team
- 5) Keeping an eye on recent industry and business trends.

1.8 Pros and Cons of Data Science

Pros	Cons
A highly demanding course in terms of job prospects.	Data science is blurry and there are no learning limits in this career.
Multi-variety of positions which makes data science course less saturated.	A data scientist requires diverse domain knowledge to become an expert.
A highly paid job in terms of career choice.	Need to be over careful about the preservation of data privacy.
Data science is versatile and is widely used in various sectors.	Data scientists need to be very careful about the inferences drawn from the analysis as wrong output may lead to failure in businesses or other sectors.
The job of data scientist is highly prestigious.	NA
The nature of job of data scientist is very interesting.	NA

Summary

- It is the task of scrutinizing and processing raw data to reach a meaningful conclusion.
- Data science is umbrella term and data analytics is a subset of data science.
- In descriptive analysis, main emphasis is on “What has happened?” by analyzing valuable information found from the available past data.
- In diagnostic analysis, the main emphasis is not only on ‘what has happened?’ but also on ‘why it happened?’.
- In predictive analysis, the main emphasis is on what might happen in near future.
- In prescriptive analysis, the recommendations are based on forecast.

Keywords

- **Nominal Data:** It contains a set of items that can be distinguished by the name of category.
- **Ordinal Data:** It contains the items that can be ordered, such as military rank, or units of government, but whose degree of difference can’t be measured.
- **Interval Data:** It contains the items that have a measurable distance between them, but no meaningful zero point, such as Fahrenheit and Celsius temperatures.
- **Ratio Data:** The measurements that have a meaningful zero and can be divided meaningfully, such as the Kelvin temperature scale.
- **Model Building:** Building a model in which team works for deploying datasets for training and testing as well as for production purposes.
- **Data visualization:** It involves graphical representation of data for easier analysis and better understanding.

Self Assessment

1. Which of these is the simplest type of data analysis?
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

2. Not only on 'what has happened?' but also on 'why it happened?'. This describes
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

3. It deals with the prediction of future based upon the available current and past data.
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

4. Which one is the most complex in this?
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

5. Which skills are required to be an analytics manager?
 - A. Python
 - B. SQL
 - C. SAS
 - D. All of the above

6. Data science is
 - A. Scrutinizing of raw data
 - B. Processing of raw data
 - C. Both of the above
 - D. None of the above

7. What are the major sources of data for studying data science?
 - A. Primary data
 - B. Secondary data
 - C. Both of the above
 - D. None of the above

8. The popular websites for data downloading are
 - A. Kaggle datasets
 - B. UCI machine learning repositories
 - C. IMDB datasets
 - D. All of the above

9. In which areas, data science can be applied?
 - A. E-commerce
 - B. Healthcare
 - C. Human resource management
 - D. All of the above

10. Data science is and data analytics is its
 - A. Umbrella term, subset
 - B. Subset, umbrella
 - C. Umbrella term, umbrella term
 - D. Subset, subset

Data Science Toolbox

11. Which of these is the simplest type of data analysis?
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

12. Not only on 'what has happened?' but also on 'why it happened?'. This describes
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

13. It deals with the prediction of future based upon the available current and past data.
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

14. Which one is the most complex in this?
 - A. Descriptive analysis
 - B. Diagnostic analysis
 - C. Prescriptive analysis
 - D. Predictive analysis

15. Which skills are required to be an analytics manager?
 - A. Python
 - B. SQL
 - C. SAS
 - D. All of the above

Answer for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. A | 2. B | 3. C | 4. D | 5. D |
| 6. D | 7. D | 8. D | 9. D | 10. A |
| 11. A | 12. C | 13. A | 14. C | 15. B |

Review Questions

1. What is data science? Explain its need. What are two major sources of data?
2. Explain the reasons why one should learn data science? Explain its use in different areas.
3. What is data analytics lifecycle? Explain its phases.
4. What are the types of data analysis? Explain.
5. What are the types of jobs in data analytics? Explain.
6. What are the pros and cons of data science? Explain in detail.

**Further Readings**

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>

**Web Links**

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Unit 02: Data Pre-Processing

CONTENTS

Objectives

Introduction

2.1 Phases of Data Preparation

2.2 Data Types and Forms

2.3 Categorical Data

2.4 Numerical Data

2.5 Hierarchy of Data Types

2.6 Possible Error Data Types

Summary

Keywords

Self Assessment

Answer for Self Assessment

Review Questions:

Further Readings

Objectives

After studying this unit, you will be able to

- understand the first concept of data science,
- understand the concept of data pre-processing,
- understand the various types of data,
- understand the possible types of errors.

Introduction

The data is often incomplete, unreliable, error prone and deficient in certain trends. The data can be

- Incomplete: Values of some attributes in data are missing.
- Noisy: Data contains error or outliers.
- Inconsistent: Data containing discrepancies in codes or names.

The data pre-processing is an essential step that needs to be considered before any analysis of data. The raw data collected from various sources needs to be pre-processed for converting it to a form that can be used for analysis. Data preparation takes place in usually two phases for any data science or data analysis project. The data preparation process phases are data pre-processing and data wrangling.

2.1 Phases of Data Preparation

Data Pre-processing: It is the task of transforming raw data to be ready to be fed into an algorithm. It is a time consuming yet important step that cannot be avoided for the accuracy of results in data analysis.

Data Wrangling: It is the task of converting data into a feasible format that is suitable for the consumption of the data. It is also known as data munging and it typically follows a set of common

steps such as extracting data from various data sources, parsing data into predefined data structures and storing the converted data into a data sink for further analysis.

2.2 Data Types and Forms

It is important to know the type of data that needs to be dealt with. There are two main types of data – categorical data and numerical data. Categorical data can be further classified into two types – Nominal and Ordinal data. Numerical data can be further classified into two types – Interval and Ratio data. Interval and ratio data are parametric; and nominal and ordinal data are non-parametric. The categorization of data types can be seen in figure 1.

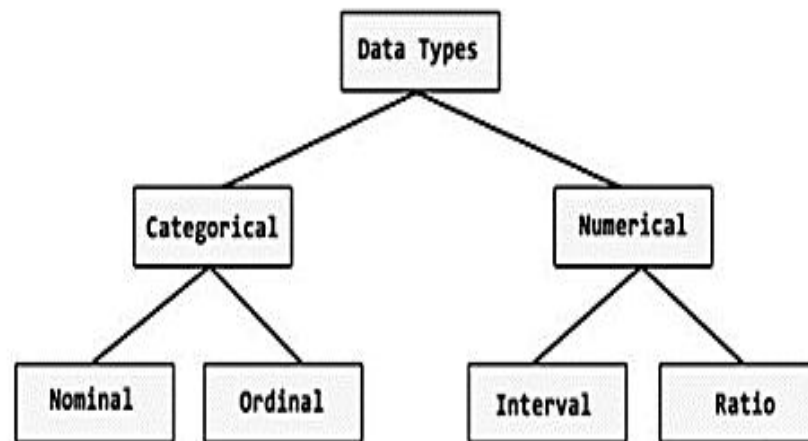


Figure 1: Categorization of data types

2.3 Categorical Data

This type of data is non-numeric and consists of text that can be coded as numeric. However, these numbers do not represent any fixed mathematical notation or meaning for the text and are simply assigned as labels or codes. It is of two types.

Nominal data: This type of data is used to label variables without providing any quantitative value. For example, gender can be labelled as 1 for Male, 2 for Female and 3 for Others. However, in real life the assigned numbers for gender are not fixed and are simply assigned for labelling. Nominal scales are mutually exclusive, that is, the values don't overlap, and the labels assigned do not bear any numerical significance.

Ordinal data: This type of data is used to label variables that need to follow some order. For example: A company may take the feedback about the quality of their service. In such a case, the possible answers could be labelled as 1 (very unsatisfied), 2 (somewhat unsatisfied), 3 (neutral), 4 (somewhat satisfied), and 5 (very satisfied). Thus, each categorical value is classified on a rating scale of 1 to 5.

2.4 Numerical Data

This type of data is numeric, and it usually follows an order of values. These quantitative data represent fixed values and can be of two types.

Interval data: This type of data follows numeric scales in which the order and exact differences between the values is considered. The interval data can be measured along a scale in which each position is equidistant from one another. The distances between each value on the interval scale are always kept equal. For example, age can be measured on an interval scale as 1, 2, 3, 4, 5 years etc. Also, income can be measured on an interval scale as Rs. 0 – 20,000, Rs. 20,001 – 40,000, Rs. 40,001 – 60,000, Rs. 60,001 – 80,000 and Rs. 80,001 – 1,00,000.

Ratio data: This type of data also follows numeric scales and has an equal and definitive ratio between each data. It is measured as multiple of one another and unlike interval data, can be multiple or divided. No negative numerical value if considered in ratio data and zero is considered as a point of origin. For instance, measurement of height and weight is an example of ratio data.

2.5 Hierarchy of Data Types

The hierarchy of data types can be seen in figure 2.

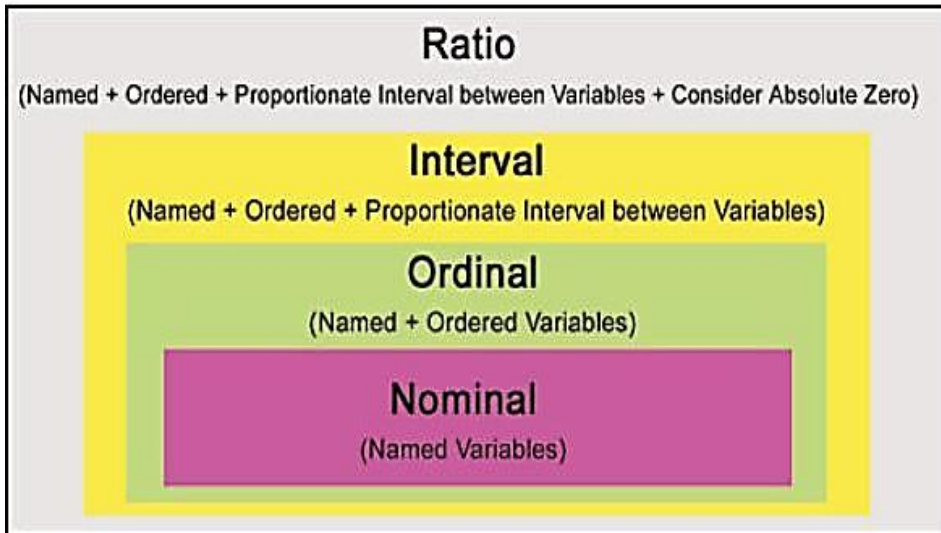


Figure 2: Hierarchy of data types

2.6 Possible Error Data Types

- 1) **Missing data:** Data is not to be filled up for various reasons. There can be various reasons for which the data can be missed. There are three types of this:
 - Missing completely at random: Forgetting to fill in the value or lost the information.
 - Missing at random data: Due to the privacy issues.
 - Missing not at random: Data may not be available.
- 2) **Manual input:** There is an error which is made in data when we are manually inputting the values that is why it is called as human made error.
- 3) **Data inconsistency:** Data may be stored in various formats. Due to different formats, the issue of data inconsistency arises.
- 4) **Wrong data types:** The data type for values is different, so the error occurs. It is different for 3 and three.
- 5) **Numerical units:** Due to varying considerations of data units. For example: weight in pounds and kilograms.
- 6) **File manipulation:** Data may be stored in CSV or text formats. Due to the varying formats, the error occurs.

Summary

- The data is often incomplete, unreliable, error prone and deficient in certain trends.
- There are two main types of data – categorical data and numerical data.
- Categorical data can be further classified into two types – Nominal and Ordinal data.

- Numerical data can be further classified into two types – Interval and Ratio data.
- The data is incomplete as the values of some attributes in data are missing, noisy as the data contains error or outliers; and inconsistent as the data containing discrepancies in codes or names.

Keywords

- **Data pre-processing:** It is the task of transforming raw data to be ready to be fed into an algorithm.
- **Data Wrangling:** It is the task of converting data into a feasible format that is suitable for the consumption of the data.
- **Categorical data:** This type of data is non-numeric and consists of text that can be coded as numeric.
- **Nominal data:** This type of data is used to label variables without providing any quantitative value.
- **Ordinal data:** This type of data is used to label variables that need to follow some order.
- **Interval data:** This type of data follows numeric scales in which the order and exact differences between the values is considered.
- **Ratio data:** This type of data also follows numeric scales and has an equal and definitive ratio between each data.

Self Assessment

1. Nominal values are
 - A. Mutual exclusive
 - B. Can be same
 - C. Not mutually exclusive
 - D. None of the above
2. Which of these techniques follows a set of common steps such as data extraction, parsing data and storing of converted data?
 - A. Data Discovery
 - B. Data Preparation
 - C. Data Wrangling
 - D. None of the above
3. Which of these types of data is non-numeric and consists of text that can be coded as numeric?
 - A. Categorical data
 - B. Numerical data
 - C. Both of the above
 - D. None of the above
4. The task of transforming raw data to be ready to be fed into an algorithm is known as
 - A. Data pre-processing
 - B. Model building
 - C. Data Wrangling
 - D. Operationalization
5. The task of converting data into a feasible format that is suitable for the consumption of the data is known as
 - A. Data pre-processing
 - B. Model building
 - C. Data Wrangling
 - D. Operationalization

6. The possible errors in missing data can be
 - A. Missing completely at random
 - B. Missing at random data
 - C. Missing not at random
 - D. All of the above

7. The data of type Named + Ordered + Proportionate Interval between Variables + Consider Absolute zero is
 - A. Ratio data
 - B. Interval data
 - C. Ordinal data
 - D. Nominal data

8. The data of Named + Ordered variables is a type of
 - A. Ratio data
 - B. Interval data
 - C. Ordinal data
 - D. Nominal data

9. Which of the following is a type of non-parametric data?
 - A. Nominal data
 - B. Ordinal data
 - C. Both of the above
 - D. None of the above

10. Which of the following is a type of parametric data?
 - A. Interval data
 - B. Ratio data
 - C. Both of the above
 - D. None of the above

11. The data preprocessing deals with
 - A. Incomplete
 - B. Noisy
 - C. Inconsistent
 - D. All of the above

12. The data preparation consists of
 - A. Data preprocessing
 - B. Data wrangling
 - C. Both of the above
 - D. None of the above

13. Which of the following is also known as data munging?
 - A. Data wrangling
 - B. Data preprocessing
 - C. Data discovery
 - D. None of the above

14. The task of converting data into a feasible format that is suitable for the consumption of the data is also known as _____.
 - A. Data Discovery
 - B. Data Preparation
 - C. Data Wrangling
 - D. None of the above

15. The data type for 5 and five will be same. This statement is
 - A. True
 - B. False

Answer for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. A | 2. C | 3. A | 4. A | 5. C |
| 6. D | 7. A | 8. C | 9. C | 10. C |
| 11. D | 12. C | 13. A | 14. C | 15. B |

Review Questions:

1. What is data pre-processing? Explain its two phases.
2. What are two main types of data? Also explain its further categorization.
3. What are the possible error data types? Explain with example.
4. What is the hierarchy of data types? Explain with examples.
5. What is data pre-processing and data wrangling? Explain in detail.



Further Readings

<https://byjus.com/maths/types-of-data-in-statistics/>



Web links

<https://www.asiapremierbpo.com/insights/4-common-data-entry-errors-and-how-to-fix-them/>

Unit 03: Various Data Pre-processing Operations

CONTENTS

Objectives

Introduction

3.1 Data Cleaning

3.2 Data Integration

3.3 Data Transformation

3.4 Data Reduction

3.5 Data Discretization

Summary

Keywords

Self Assessment

Answer for Self Assessment

Review Questions:

Further Readings

Objectives

After studying this unit, you will be able to

- understand the concept of data preprocessing,
- understand the process and methods of data cleaning,
- understand the concept of data integration,
- understand the data integration framework and tools,
- understand the concept, need and techniques of data transformation,
- understand the concept, need and strategies of data reduction
- understand the concept of data discretization.

Introduction

The raw data collected are prone to several errors and thus need to go through a series of steps for preprocessing. There are various data pre-processing operations like data cleaning, data integration, data transformation, data reduction and data discretization.

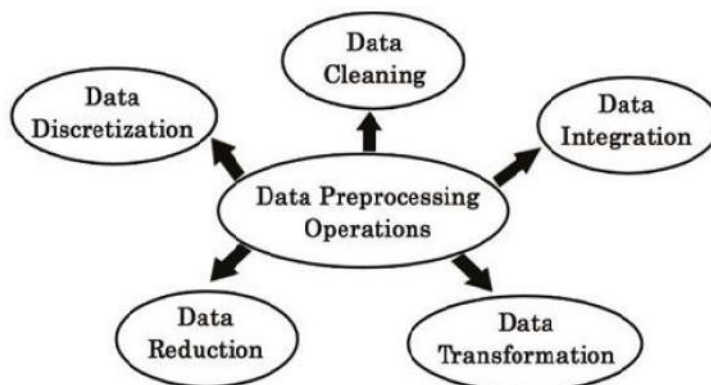


Figure 1: Data Pre-processing Operations

3.1 Data Cleaning

Dirty data can cause an error while doing data analysis. Data cleaning is done to handle irrelevant or missing data. Data is cleaned by:

- 1) Filling in the missing values.
- 2) Smoothing noisy data
- 3) Detecting and removing outliers

Filling the missing values

Filling up the missing values in data is known as the imputation of missing data. Sometimes, this imputation process becomes time-consuming. The method to be adapted for filling up the missing values depends on the pattern of data used and the nature of analysis to be performed with the data. The simplest methods used for handling missing values are:

- 1) Replace Missing Values with Zeroes
- 2) Dropping Rows with Missing Values
- 3) Replace Missing Values with Mean/Median/Mode
- 4) Filling every missing value with the previous value
- 5) Filling every missing value with the next value


Smoothing the noisy data

It is an important technique for data preprocessing. It is intended to identify trends in the presence of noisy data when the pattern of the trend is unknown. Two main methods used for smoothing noisy data are: binning and regression.

Binning

The method of binning is mainly applied to improve the accuracy of a predictive model by reducing the noise from data. It also helps in easy identification of outliers or invalid values for numerical data. It is a discretization method that performs local smoothing of data by transforming numerical values into categorical counterparts. This method smoothens sorted data values by referring to the values around it. The sorted values are distributed into a number of bins or buckets. For instance, values of age can be categorized into groups to make it fall into separate bins. One bin consists of age 10-19, another bin consists of age 20-29, the third bin consists of age 30-39; another bin consists of age 40-49, and so on. Thus, all the numerical values are discretized to form a frequency table. Distributing of values into bins can be done in two ways:

- 1) **Equal width binning:** The data is divided into n intervals of equal size. The width w of the interval is calculated as $w = (\text{maxvalue} - \text{minvalue}) / n$.
- 2) **Equal frequency binning:** The data is divided into n groups and each group contains approximately the same number of values.

	<p>Example:Data Values: 0, 3, 13, 17, 17, 19, 23, 28, 29</p> <p>Equal Width Binning:</p> <p>Bin 1: 0, 3 [-, 10]</p> <p>Bin 2: 13, 17, 17, 19 [10,20]</p> <p>Bin 3: 23, 28, 29 [20, +]</p> <p>Equal Frequency Binning:</p> <p>Bin 1: 0, 3, 13 [-, 14]</p> <p>Bin 2: 17, 17, 19 [14,21]</p>
---	--

Bin 3: 23, 28, 29 [21, +]

Binning is often used as a standard method of data smoothing to predict trends and to make a quick analysis of the ranges of the entire data. In the binning method, the bins are formed by consulting the neighborhood values. For this reason, the binning method is considered to perform local smoothing.

Regression

This method smoothens data by fitting the data to a function. The simplest type of regression is the linear regression which works by finding the "best fitted " line to fit two attributes so that the values of the independent attribute can be used to predict the values of the dependent attribute. There are various regression techniques such as linear regression, logistic regression, decision tree regression, polynomial regression, and lasso regression. For each of these types of regression, the main concept is the same, that is, to establish the nature of the functional relationship between two or more variables for future prediction or forecasting.

Detecting and removing outliers

An outlier is a data point that is far away from other related data points. Outliers may occur due to several reasons such as measurement error, data entry error, experimental error, intentional inclusion of outliers, sampling error or natural occurrence of outliers. For data analysis, outliers should be excluded from the dataset as much as possible as these outliers may mislead the analysis process resulting in incorrect results and longer training time. In turn, the model developed will be less accurate that provides comparatively poorer results.

There are several ways to detect outliers in each dataset. Few such popular methods of outlier detection are listed below:

- 1) Probabilistic and Statistical Modelling (parametric)
- 2) Z-Score or Extreme Value Analysis (parametric)
- 3) Proximity Based Models (non-parametric)
- 4) Linear Regression Models (PCA, LMS)
- 5) High Dimensional Outlier Detection Methods (high dimensional sparse data)

Statistical method of outlier detection

- 1) Standard deviation method
- 2) Interquartile range method

Standard deviation method:

This method of outlier detection initially calculates the mean and standard deviation of the data points. Each value is then compared by checking whether the value is a certain number of standard deviations away from the mean. If so, the data point is identified as an outlier. The specified number of standard deviations is considered as the threshold value for which the default value is 3.

Interquartile range method:

This method of outlier detection initially calculates the interquartile range (IQR) for the given data points. Each value is then compared with the value $(1.5 \times \text{IQR})$. If the data point is more than $(1.5 \times \text{IQR})$ above the third quartile or below the first quartile, the data point is identified as an outlier. This can be mathematically represented as low outliers are less than $Q1 - (1.5 \times \text{IQR})$, and high outliers are more than $Q3 + (1.5 \times \text{IQR})$, where $Q1$ is the first quartile and $Q3$ is the third quartile.

3.2 Data Integration

The technique of data integration allows merging data from various disparate sources to maintain a unified view of the data. It is an important technique used mainly for merging varying data of a company in a common unified format or for combining data of more than one company to maintain

common data assets. The task is complex as the data sources in real life are heterogeneous and this raises the complexity of assimilating the data of different formats into a common format to be stored in a unified data source.

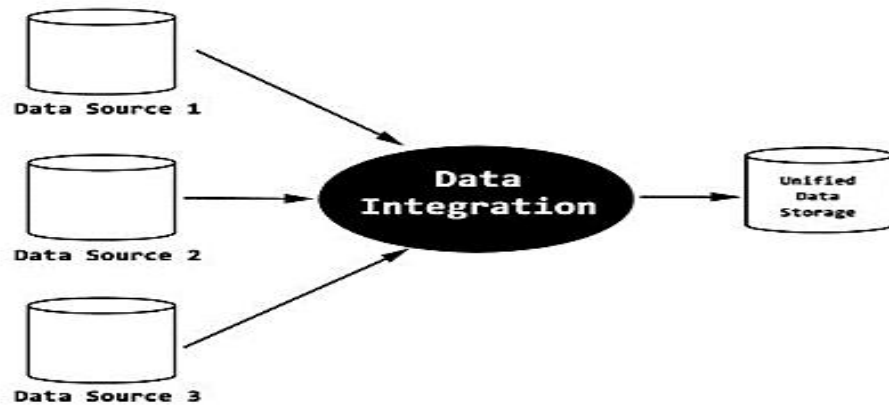


Figure 2: Data Integration

Data integration is carried out in many areas such as data warehousing, data migration, information integration, and enterprise management. It is a challenging work as a lot of understanding of the system is required prior to integrating data from multiple sources, such as:

- 1) What is the main objective of the integration of data for the company?
- 2) What kind of data assets are to be integrated?
- 3) What are the various sources of data?
- 4) What are the various business rules to be followed?
- 5) What is the support model for the new system?
- 6) Who will be the owner of the entire system?

One important aspect to consider for data integration is to handle redundancy. Redundant data can be detected using the concept of correlation analysis. There are several methods used in correlation analysis to find the correlation coefficient (a value between 0 and 1), which measures the strength and the direction of a linear relationship between two variables. One of the standard methods of finding the correlation coefficient is called Karl Pearson's Coefficient of Correlation (denoted by r), which can be mathematically represented as:

$$r = \frac{\sum xy}{N\sigma_x\sigma_y}$$

Standard Data Integration Techniques

Virtual integration

This technique, also called Uniform Data Access, lets the data be viewed from a single source system, but the entire data is not stored in a single location. Data virtualization can provide a unified view of data to customers across a platform without maintaining a separate store for the consolidated data. However, if this method is used, there is a limited possibility of accessing data history and version management.

Physical data integration

This technique, also called Common Data Storage, maintains a copy of the data from the source to a new system. Here, the collected integrated data from several sources is stored and managed by the new system instead of the source system. This approach is typically followed by data warehouse systems.

Application-based integration

This technique takes the help of application(s) to implement the integration process of collecting and storing data in a unified single storage system. This technique has limited use compared to physical data integration.

Unit 03: Various Data Preprocessing Operations

Manual integration

This technique, also called as Common User Interface, is largely used for accessing information that is available on the Internet. Here, the entire data is not stored in a single location for unified data access; rather the source system is used to access the required web pages.

Middleware data integration

This technique handovers the integration process from particular applications to a new middleware layer. The applications do not fully implement the integration logic but partially participate in the data integration.

Data integration framework

The data integration framework (DIF) involves two categories of processes as given below:

- 1) The first category involves the process of determining the data requirements and solutions. Data requirements can be -studying business requirements, determining data and quality needs, data profiling, performing a data quality assessment based on business requirements, and so on.
- 2) The second category involves the process used to physically accumulate the data from multiple sources and transform it into meaningful information to be used for analysis and decision making.

For carrying out this process, the various steps involved are:

- **Data preparation:** This includes gathering, reformatting, combining, transforming, cleansing and storing data for further analysis.
- **Data franchising :** This involves reconstructing data into information that can be used for reporting and analysis.
- **Data and metadata management :** This involves managing both the data and metadata between the several processes.

Data Integration Tools

Data Integration Tool	Data Integration Capabilities/Services	Main feature(s)
Informatica	Provides advanced hybrid data integration	Integrated codeless environment
Microsoft	Provides hybrid data integration, provides its own Server Integration Services	Fully managed ETL service in the cloud.
Talend	Integrates data with unified development and management tools	Open, scalable architectures; Five times faster than MapReduce
Oracle	Cloud-based data integration	Machine learning and AI capabilities; data migration across hybrid environments; data profiling and governance
IBM	Data integration for structured and unstructured data	Massive parallel processing capabilities; data profiling, standardization, machine enrichment.

Other standard integration tools

- SAP
- Information builders
- SAS
- Adeptia
- Actian
- Dell Bhoomi
- Syncsort

These integration tools are mainly involved in solving various complex data integration processes such as ingestion, cleansing, ETL mapping and transformation.

3.3 Data Transformation

Once the data is cleaned and integrated, it is transformed into a range of values that are easier to be analyzed. This is done as the values for different information are found to be in a varied range of scales. For example, for a company, age values for employee scan be within the range of 20-55 years whereas salary values for employees can be within the range of Rs. 10,000 - Rs. 1,00,000. This indicates one column in a dataset can be more weighted compared to others due to the varying range of values. In such cases, applying statistical measures for data analysis across this dataset may lead to unnatural or incorrect results. Data transformation is hence required to solve this issue before applying any analysis of data.

Various data transformation techniques

- 1) Rescaling data
- 2) Normalizing data
- 3) Binarizing data
- 4) Standardizing data
- 5) Label encoding
- 6) One hot encoding

Rescaling data

When the data encompasses attributes with varying scales, many statistical or machine learning techniques prefer rescaling the attributes to fall within a given scale. Rescaling of data allows scaling all data values to lie between a specified minimum and maximum value (say, between 0 and 1). Data rescaling is done prior to data analysis in many cases such as, in algorithms that weight inputs like regression and neural networks, in optimization algorithms used in machine learning, and in algorithms that use distance measures like K-Nearest Neighbors.

Normalizing data

Normalizing rescales data in such a way that each row of observation equals to a length of 1 (called a unit norm in linear algebra). Data normalization is done prior to data analysis in many cases such as for sparse data having lots of zeroes or for attributes having high varying ranges of data values.

Binarizing data

Binarizing is the process of converting data to either 0 or 1 based on a threshold value. All the data values above the threshold value are marked 1 whereas all the data values equal to or below the threshold value are marked as 0. Data binarizing is done prior to data analysis in many cases such as, dealing with crisp values for the handling of probabilities and adding new meaningful features in the dataset.

Standardizing data

Standardization, also called mean removal, is the process of transforming attributes having a Gaussian distribution with differing mean and standard deviation values into a standard Gaussian distribution with a mean of 0 and a standard deviation of 1. Standardization of data is done prior to

Unit 03: Various Data Preprocessing Operations

data analysis in many cases such as, in the case of linear discriminate analysis, linear regression, and logistic regression.

Label encoding

The process of label encoding is used to convert textual labels into numeric form in order to prepare it to be used in a machine-readable form. The labels are assigned a value of 0 to (n-1) where n is the number of distinct values for a particular categorical feature. The numeric values are repeated for the same label of that attribute. For instance, let us consider the feature 'gender' having two values - male and female. Using label encoding, each gender value will be marked with unique numerical values starting with 0. Thus, males will be marked 0, females will be marked 1.

One hot encoding

It refers to splitting the column which contains numerical categorical data to many columns depending on the number of categories present in that column. Each column contains "0" or "1" corresponding to which column it has been placed. For instance, let us consider a feature X has two distinct values A and B as follows: X: A A B. If one hot encoding is applied to the above feature, the matrix that will be formed as follows:

1 [1,0]

2 [1,0]

3[0,1]

3.4 Data Reduction

An essential step of data preprocessing is data reduction. To reduce the unimportant or unwanted features from a dataset.

Strategies for data reduction

- 1) Dimensionality reduction
- 2) Data cube aggregation
- 3) Numerosity reduction

Dimensionality Reduction

For classification and clustering, features are studied to obtain the analysis output. The higher the number of features, the more difficult it is for the training process for output analysis. There may be many features that are correlated or redundant. The main aim is to obtain a set of principal variables. There are two main components:

- 1) Feature selection
- 2) Feature extraction

Feature selection

The main aim is to obtain a smaller subset that can be used to model a given problem. It usually involves three ways - filter, wrapper, and embedded. The standard techniques for feature selection are:

- 1) Univariate selection: This method works by inspecting each feature and then finding the best feature based on statistical tests. It also analyses the capability of these features in accordance with the response variable.
- 2) Recursive feature elimination: This method works by performing a greedy search to acquire the best feature subset from a given dataset. This is done in an iterative process by determining the best or the worst feature at each iteration.
- 3) Stepwise forward selection : This method initially starts with an empty set of attributes which is considered as the minimal set. In each iteration, the most relevant attribute is then added to the minimal set until the stopping rule is satisfied. One of the stopping rules is to stop when all remaining variables have a p-value above some threshold.
- 4) Stepwise backward elimination: This method initially starts with all the sets of attributes that are considered as the initial set. In each iteration, the most irrelevant attribute is then removed from the minimal set until the stopping rule is satisfied. One of the stopping rules is to stop when all remaining variables have a significant p-value defined by some significance threshold.

- 5) Combination of forward selection and backward elimination: This method is commonly used for attribute subset selection and works by combining both the methods of forward selection and backward elimination.
- 6) Decision tree induction: This method uses the concept of decision trees for attribute selection. A decision tree consists of several nodes that have branches. The nodes of a decision tree indicate a test applied on an attribute while the branch indicates the outcome of the test. The decision tree helps in discarding the irrelevant attributes by considering those attributes that are not a part of the tree.

Feature extraction

This process is used to reduce the data in a high dimensional space to a lower dimension space. While feature selection chooses the most relevant features from among a set of given features, feature extraction creates a new, smaller set of features that consists of the most useful information. Few methods of dimensionality reduction

- 1) PCA
- 2) LDA

Principal Component Analysis

It is an unsupervised method of feature extraction that creates linear combinations of the original features. The features are uncorrelated and are ranked in order of variance. The data has to be normalized before performing PCA. PCA has several variations of it such as sparse PCA, kernel PCA, and so on.

Linear Discriminant Analysis

It is a supervised method of feature extraction that also creates linear combinations of the original features. However, it can be used for only labeled data and can be thus used only in certain situations. The data has to be normalized before performing LDA. This helps in removing redundant features, reducing computation time, as well as in reducing storage space. However, dimensionality reduction results in loss of data and should be used with proper understanding to effectively carry out data preprocessing before performing analysis of data.

Data Cube Aggregation

Data cubes store multidimensional aggregated information. For example, a sales report has been prepared to analyze the number of sales of mobile phones per brand in each branch for the year 2009 to 2019.

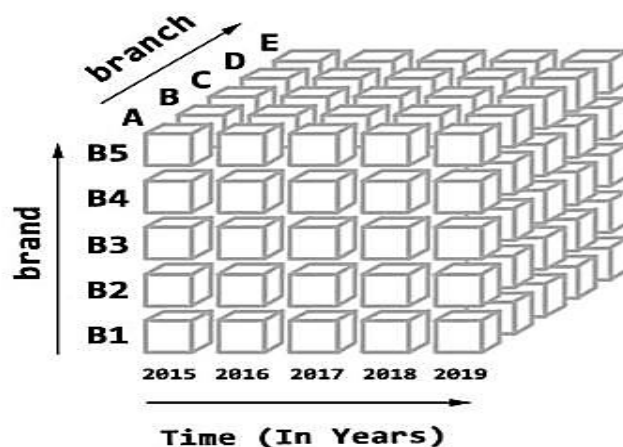


Figure 3: Data cube

This has three dimensions - time, brand and branch. All the dimensions together help to solve queries. Thus, OLAP cubes are designed using business logic and understanding. They are optimized for analytical purposes so that they can report on millions of records at a time.

Cell in Data Cube

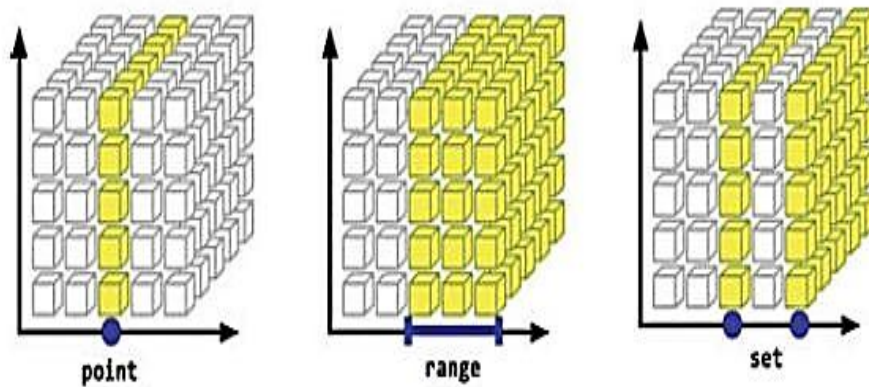


Figure 4: A point, range and set

Cubes are used in many ways for several tasks, such as:

- Model preparation
- Measure aggregation
- Drill-down through dimensions
- Slice-and-dice

Numerosity Reduction

This method is used for converting the data to smaller forms to reduce the volume of data. Numerosity reduction may be either parametric or non-parametric. Parametric methods use a model to represent data in which parameters of the data are stored, rather than the data itself. Examples of parametric models include regression and log-linear models. Non-parametric methods are used for storing reduced representations of the data. Examples of non-parametric models include clustering, histograms, sampling, and data cube aggregation.

3.5 Data Discretization

The data discretization method is used to partition the range of continuous attributes into intervals. By doing so, several continuous attribute values are replaced by lesser interval labels. Data discretization can be of two types - top-down discretization and bottom-up discretization:

- Top-down discretization: This process, also called splitting, works by first finding one or a few split points to divide the entire attribute range. This process is repeated on the resulting intervals until the stopping condition is found to be true.
- Bottom-up discretization: This process, also called as merging, works by first considering all the continuous values as potential split-points. The process then further removes some continuous values by merging neighborhood values to form intervals.

Concept hierarchy: If an attribute is taken into consideration and the technique of discretization is performed recursively on the attribute to provide a hierarchical partitioning of its values, it is known as a concept hierarchy. Often the clustering technique is used to generate a concept hierarchy by considering either splitting process or the merging process. Clustering using the splitting process initially decomposes each cluster or partition into subclusters. This forms a lower level of the hierarchy of clusters. In the merging process of clustering for discretization, clusters are created by recurrently grouping neighboring clusters to form higher-level concepts.

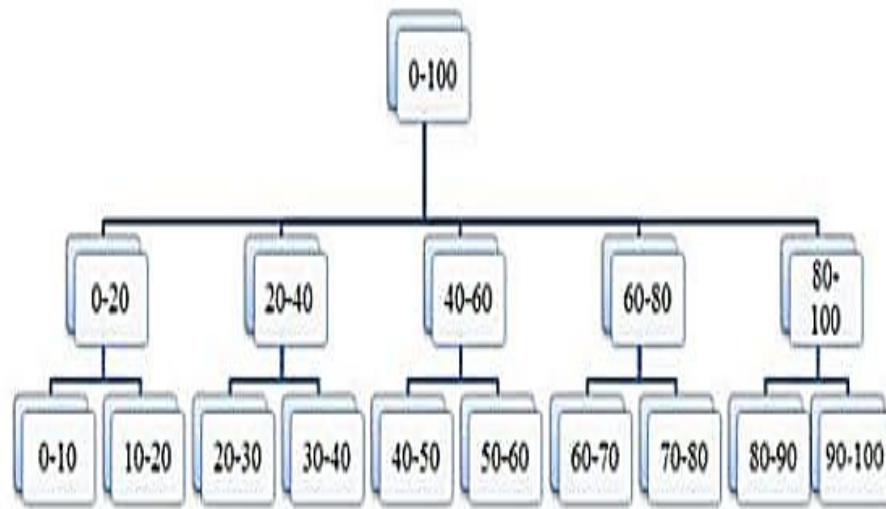


Figure 5: Concept hierarchy

Summary

- Data cleaning is done to handle irrelevant or missing data.
- Data is cleaned by filling in the missing values, smoothing noisy data and detecting and removing outliers.
- Binning is often used as a standard method of data smoothing to predict trends and to make a quick analysis of the ranges of the entire data.
- In Karl Pearson Coefficient, the value of $r = +1$ indicates a perfect positive relationship between two variables; $r = -1$ indicates a perfect negative relationship between two variables, and $r=0$ indicates that there is no relationship between the two variables.
- Once the data is cleaned and integrated, it is transformed into a range of values that are easier to be analyzed. This is done as the values for different information are found to be in a varied range of scales.
- If an attribute is taken into consideration and the technique of discretization is performed recursively on the attribute to provide a hierarchical partitioning of its values, it is known as a concept hierarchy.

Keywords

- **Imputation of missing data:** Filling up the missing values in data is known as the imputation of missing data.
- **Binning:** It is a discretization method that performs local smoothing of data by transforming numerical values into categorical counterparts.
- **Equal width binning:** The data is divided into n intervals of equal size. The width w of the interval is calculated as $w = (\text{maxvalue} - \text{minvalue}) / n$.
- **Equal frequency binning:** The data is divided into n groups and each group contains approximately the same number of values.
- **Outlier:** An outlier is a data point that is far away from other related data points.
- **Interquartile range method:** It initially calculates the interquartile range (IQR) for the given data points. Each value is then compared with the value $(1.5 \times \text{IQR})$. If the data point is more than $(1.5 \times \text{IQR})$ above the third quartile or below the first quartile, the data point is identified as an outlier.
- **Data Integration:** The technique of data integration allows merging data from various disparate sources so as to maintain a unified view of the data.
- **Data franchising :** This involves reconstructing data into information that can be used for reporting and analysis.

Unit 03: Various Data Preprocessing Operations

- **Data and metadata management** : This involves managing both the data and metadata between the several processes.
- **Rescaling**: Rescaling of data allows scaling all data values to lie between a specified minimum and maximum value (say, between 0 and 1).
- **Binarizing**: Binarizing is the process of converting data to either 0 or 1 based on a threshold value.
- **Standardization**: Standardization, also called mean removal, is the process of transforming attributes having a Gaussian distribution with differing mean and standard deviation values into a standard Gaussian distribution with a mean of 0 and a standard deviation of 1.
- **Label encoding**: The process of label encoding is used to convert textual labels into numeric form to prepare it to be used in a machine-readable form.
- **Feature Selection**: The main aim is to obtain a smaller subset that can be used to model a given problem. It usually involves three ways - filter, wrapper, and embedded.
- **Data Discretization**: The data discretization method is used to partition the range of continuous attributes into intervals. By doing so, several continuous attribute values are replaced by lesser interval labels. Data discretization can be of two types - top-down discretization and bottom-up discretization.

SelfAssessment

1. Which of these represents the techniques of data transformation?
 - A. Binarizing data
 - B. Label encoding
 - C. One hot encoding
 - D. All of the above

2. _____ is the process of converting data to either 0 or 1 based on a threshold value.
 - A. Binarizing data
 - B. Label encoding
 - C. One hot encoding
 - D. Standardizing data

3. The process of _____ is used to convert textual labels into numeric form in order to prepare it to be used in a machine-readable form.
 - A. Binarizing data
 - B. Label encoding
 - C. One hot encoding
 - D. Standardizing data

4. Where is the data integration carried out?
 - A. Data warehousing
 - B. Data migration
 - C. Information integration
 - D. All of the above

5. The redundant data can be detected using
 - A. Concept of correlation analysis
 - B. Concept of regression analysis
 - C. Concept of classification analysis
 - D. Concept of clustering analysis

6. The value +1 of correlation coefficient indicates
 - A. No relationship
 - B. A perfect positive relationship
 - C. A perfect negative relationship
 - D. Just a number

7. The value -1 of correlation coefficient indicates
 - A. No relationship
 - B. A perfect positive relationship
 - C. A perfect negative relationship

- D. Just a number
8. Which of these techniques of data integration is also known as Uniform Data Access?
A. Virtual integration
B. Physical data integration
C. Application based integration
D. Manual integration
9. Which of these are data integration tools?
A. Microsoft
B. Talend
C. IBM
D. All of the above
10. Which of the following are data pre-processing operations?
A. Data reduction
B. Data cleaning
C. Data transformation
D. All of the above
11. The data is cleaned by:
A. Filling in the missing values
B. Smoothing noisy data
C. Detecting and removing outliers
D. All of the above
12. The methods for smoothing noisy data are:
A. Regression
B. Binning
C. Both of the above
D. None of the above
13. Which of these can be used for smoothing of noisy data?
A. Equal width binning
B. Equal frequency binning
C. Both of the above
D. None of the above
14. _____ works by performing a greedy search to acquire the best feature subset from a given dataset.
A. Recursive feature elimination
B. Univariate selection
C. Stepwise forward selection
D. Stepwise backward elimination
15. Which process of discretization is also known as splitting?
A. Top-down discretization
B. Bottom-up discretization
C. Left-right discretization
D. Right-left discretization

Answer for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. D | 2. A | 3. B | 4. D | 5. A |
| 6. B | 7. C | 8. A | 9. D | 10. D |
| 11. D | 12. C | 13. C | 14. A | 15. A |

Review Questions

1. What is data pre-processing? Explain its different operations in detail.
2. What is data cleaning? What is the need of data cleaning? Explain the strategies by which we can clean the data.
3. In data cleaning, explain how can we detect and remove the outliers?
4. What is data integration? How can we handle redundancies?
5. What are standard data integration techniques? Explain.
6. What is data integration framework? Explain its phases.
7. What is data transformation? Explain various data transformation techniques.
8. What is data reduction? What are different strategies for data reduction?
9. What are two main strategies for dimensionality reduction? Explain.

**Further Readings**

<https://www.javatpoint.com/data-preprocessing-machine-learning>

**Web Links**

<https://www.geeksforgeeks.org/data-cleansing-introduction/>

Unit 04: Data Plotting and Visualization

CONTENTS

Objectives

Introduction

4.1 Data visualization

4.2 Visual Encoding

4.3 Concepts of Visualization Graph

4.4 Role of Data Visualization and its Corresponding Visualization Tool

4.5 Data Visualization Softwares

4.6 Data Visualization Libraries

4.7 Matplotlib Library

4.8 Advanced Data Visualization using Seaborn Library

4.9 Visualization Libraries

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this unit, you will be able to:

- Understand the concept of data visualization
- Understand the importance of data visualization
- Understand the data visualization softwares and libraries
- Understand the advanced visualization using Seaborn library
- Understand the types of data visualization

Introduction

To make the information easy to understand and analyze, we use the concept of visualization which is a graphical way to represent the data. Data visualization has the capacity of demonstrating the complex data relationships and patterns with the help of simple designs consisting of lines, shapes, and colors.

4.1 Data visualization

It is considered as an evolving blend of art and science that has brought a revolutionary change in the corporate sectors and will also continue to do so over the next few years. It plays a major role in the decision making of the analytics world. Is the process of presenting data in the form of graphs or charts?

- **Visualize:** To form a mental vision, image, or picture of something to make visible to the mind or imagination.

- **Visualization:** It is the use of computer graphics to create visual images which aid in the understanding of complex, often massive representations of data.
- **Visual Data Mining:** It is the process of discovering implicit but useful knowledge from large data sets using visualization techniques.

Table vs Graph

- A table is best when you need to look up specific values. A graph is best when the message is contained in the shape of the values.
- A table is best when users need precise values and there is a need to precisely compare related values. A graph is best when you want to reveal relationships among multiple values (similarities and differences)

Data Visualization can help in

- **Identify outliers in data:** An outlier is a data point that is aloof or far away from other related data points. Outliers may occur due to several reasons such as measurement error, data entry error, experimental error, intentional inclusion of outliers, sampling error or natural occurrence of outliers. For data analysis, outliers may mislead the analysis process resulting in abruptly different, incorrect results and longer training time.
- **Improve response time:** Data visualization gives a quick glance of the entire data and, in turn, allows analysts or scientists to quickly identify issues, thereby improving response time. This contrasts with huge chunks of information that may be displayed in textual or tabular format covering multiple lines or records.
- **Greater simplicity:** Data, when displayed graphically in a concise format, allows analysts or scientists to examine the picture easily. The data to be concentrated on gets simplified as analysts or scientists interact only with relevant data.
- **Easier visualization of patterns:** Data presented graphically permits analysts to effortlessly understand the content for identifying new patterns and trends that are otherwise almost impossible to analyze. Trend analysis or time-series analysis are in huge demand in the market for a continuous study of trends in the stock market, companies, or business sectors.
- **Business analysis made easy:** Business analysts can deal with various important decision-making such as sales prediction, product promotion, and customer behavior using correct data visualization techniques.
- **Enhanced collaboration:** Advanced visualization tools make it easier for teams to collaboratively go through the reports for instant decision-making.

Advantages of Data Visualization

- It helps to understand large and complex amounts of data very easily.
- It allows the decision-makers to make decisions very efficiently and also allows them in identifying new trends and patterns very easily.
- It is also used in high-level data analysis for Machine Learning and Exploratory Data Analysis (EDA).

4.2 Visual Encoding

Visual encoding is the approach used to map data into visual structures, thereby building an image on the screen. This is an important consideration to be made as one visualization tool can be more effective than the other visualization tool due to the easy perception of information conveyed by the former visualization graph than the latter. The attribute values signify important data characteristics such as numerical data, categorical data, or ordinal data. Spatiotemporal data contains special attributes such as geographical location (spatial dimension) and/or time (temporal dimension). Mapping of data is based on the visual cues (also called retinal variables) such as location, size, color value, color hue, color saturation, transparency, shape, structure, orientation, and so on.

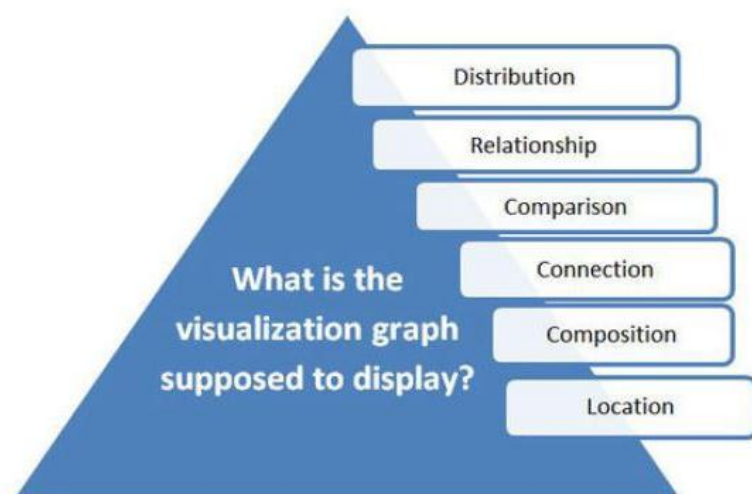
Retinal Variables

To represent data that involves three or more variables, these retinal variables play a major role.

- **Size:** It is an important visualizer for quantitative data as a smaller size indicates less value while bigger size indicates more value.
- **Color hue:** It plays an important role in data visualization as for instance, the red color signifies something alarming, the blue color signifies something serene and peaceful, while the yellow color signifies something bright and attractive.
- **Shape:** Such as circle, oval, diamond and rectangle, may signify different types of data and is easily recognized by the eye for the distinguished look.
- **Orientation:** Such as vertical, horizontal and slanted, help in signifying data trends such as an upward trend or a downward trend.
- **Color Saturation:** This decides the intensity of color and can be used to differentiate visual elements from their surroundings by displaying different scales of values.
- **Length:** It decides the proportion of data and is a good visualization parameter for comparing data of varying values.

4.3 Concepts of Visualization Graph

The main question for visual encoding lies in the fact 'What do we want to portray with the given data?'



4.4 Role of Data Visualization and its Corresponding Visualization Tool

Role of Data Visualization	Data Visualization Graph
Distribution	Scatter Chart
	3D Area Chart
	Histogram
Relationship	Bubble Chart
	Scatter Chart

Comparison	Bar Chart
	Column Chart
	Line Chart
	Area Chart
Composition	Pie Chart
	Waterfall Chart
	Stacked Column Chart
	Stacked Area Chart
Location	Bubble Map
	Choropleth Map
	Connection Map
Connection	Matrix Chart
	Node-link Diagram
	Word Cloud
	Alluvial Diagram
	Tube Map

4.5 Data Visualization Softwares

These software applications differ in their ability to use various types of graphs and visuals, simplified user interface, accurate trend tracking capability, level of security, ease of use in mobile and friendly report generation.

Data Visualization Software	Description	Key Features
Tableau	Easily connects, visualizes, and shares data with effective seamless experience from desktop to mobile	Flexible data analysis methods Mobile friendly Manageable permission access
Qlikview	Allows users to create default and custom data connectors and templates, depending on one's need	Personalized data search Script building Role-based access
Sisense	Uses agile analysis software with a variety of data visualization options. Can create dashboards and graphics with drag and drop user	Easy setup and use, Data exporting range, Interactive dashboards

Unit 04: Data Plotting and Visualization

	interface.	
Looker	It provides an efficient business intelligence platform for users to utilize SQL for organizing unstructured data.	Ease of use Strong collaboration features Handy and compact visualization.
Zoho Analytics	Uses a variety of tools, such as pivot tables, KPI widgets, and tabular view components. Can generate reports with valuable business insights.	Insightful reports, Robust security, Integration and app development
Domo	Generates real time data in a single dashboard. Can generate various creative data displays such as multi-part widgets and trend indicators	Free trial, Socialization, Dashboard creation
Microsoft Power BI	Comes with unlimited access to on-site and in-cloud data that gives a centralized data access hub.	Unlimited connective options, Affordability, Web publishing
IBM Watson Analytics	Can type in various questions which the intelligence software can interpret and answer accordingly	File Upload Public forum support On-site data
SAP Analytics Cloud	Can generate focused reports and collaborative tools for online discussion. Provides import and export features for spreadsheets and visuals	Easy Forecasting Set up important events Cloud-based protection
Plotly	Provides a vast variety of colorful designs for data visualization. Can use the chart studio to create web-based reporting templates	2D and 3D Chart options, Open-source coding, Designer input

Few Other Softwares

Few other softwares available for plotting visualization graphs:

1. MATLAB
2. FusionCharts
3. Datawrapper
4. Periscope Data
5. Klipfolio
6. Kibana
7. Chartio
8. Highcharts
9. Infogram

4.6 Data Visualization Libraries

The libraries which are very important for data visualization are:

1. The matplotlib library
2. The seaborn library
3. The ggplot library
4. The Bokeh library
5. The plotly library
6. The pygal library
7. The geoplotlib library
8. The Gleam library
9. The missingno library
10. The Leather library

The matplotlib library

It is the most common and standard Python library used for plotting 2D data visualizations. This library was created by John D. Hunter and is currently maintained by a team of Python developers. The matplotlib library is mainly used for creating plots that can be zoomed in on a section of the plot and pan around the plot using the toolbar in the plot window. It is the first data visualization library to be developed in Python, and later many other libraries were built on top of it for various other ways of visualizations. This library is used to create a variety of visualization graphs such as line plots, pie charts, scatter plots, bar charts, histograms, stem plots, and spectrograms. The matplotlib allows easy use of labels, axes titles, grids, legends, and other graphic requirements with customizable values and text. Matplotlib is a low-level library of Python which is used for data visualization. It is easy to use and emulates MATLAB like graphs and visualization. This library is built on the top of NumPy arrays and consist of several plots like line chart, bar chart, histogram, etc. It provides a lot of flexibility but at the cost of writing more code.

The sea born library

The sea born library couples the power of the matplotlib library to create artistic charts with very few lines of code. The sea born library follows creative styles and rich color palettes, that allows to create visualization plots to be more attractive and modern. Today's visualization graph is mainly plotted in sea born rather than matplotlib, primarily because of the sea born library's rich color palettes and graphic styles that is much more stylish and sophisticated than matplotlib. As sea born is considered to be a higher-level library, there are certain special visualization tools such as, violin plots, heat maps and time series plots that can be created using this library.

The ggplot library

The ggplot library is based on the ggplot2 library which is an R plotting system and concepts are based on the Grammar of Graphics. The ggplot library creates a layer of components for creating plots which makes it different from matplotlib based on the operations of plotting graph. It is integrated with pandas and is mainly used for creating very simple graphics. This library sacrifices the complexity of plotting complex graphs as its primary focus is on building basic graphs that are often required for analyzing simple data distribution.

The Bokeh library

The Bokeh library is native to Python and is mainly used to create interactive, web-ready plots, which can be easily output as HTML documents, JSON objects, or interactive web applications. Like ggplot, its concepts are also based on the Grammar of Graphics. It has an added advantage of managing real-time data and streaming. This library can be used for creating common charts such as histograms, bar plots, and box plots. It can also handle very minute points of a graph such as handling a dot of a scatter plot. Using Bokeh, it is easy for a user to control and define every element of the chart without using any default values and designs. There are three varying interfaces supported by Bokeh for being used by different types of users. The topmost level interface allows building simple, common charts and graphs very easily. The middle-level interface allows designing visualization graphs with a provision to control the basic building blocks of each graph. Lastly, the bottom level interface is used mainly by software engineers and developers as this

interface provides full support for controlling and customizing each and every component of a graph to deal with complex graphics.

The Plotly library

The plotly library is an online platform for data visualization and it can be used in making interactive plots that are not possible using other Python libraries. Few such plots include dendrograms, contour plots, and 3D charts. Other than these graphics, some basic visualization graphs such as area charts, bar charts, box plots, histograms, polar charts, and bubble charts can also be created using the plotly library. One interesting fact about plotly is that the graphs are not saved as images but rather serialized as JSON, because of which the graphs can be opened and viewed with other applications such as R, Julia, and MATLAB.

The Pygal library

The pygal library creates interactive plots that can be embedded in the web browser. It also has the ability to output charts as Scalable Vector Graphics (SVGs). All the chart types created using pygal are packaged into a method that makes it easy to create an artistic chart in a few lines of code. For instance, to create a bar chart, simply import the pygal library and then create a variable to assign the value of pygal.Bar(). The graph created can finally be saved in .svg extension to get a stylized CSS formatting.

The Geoplotlib library

The geoplotlib is a toolbox for designing maps and plotting geographical data. Few of the map-types that can be created are heat maps, dotdensity maps, and choropleths. In order to use geoplotlib, one has to also install Pyglet, an object-oriented programming interface. This library is mainly used for drawing maps as no other Python libraries are meant for creating graphics for maps.

The Gleam library

The Gleam library is like the R's Shiny package. It converts analyses into interactive web apps using Python scripts. It creates a web interface that lets anyone play with the data in real-time. One interesting capability of this library is that fields can be created on top of the graphic and users can filter and sort data by choosing appropriate field.

The missingno library

The missingno library can deal with missing data and can quickly measure the wholeness of a dataset with a visual summary, instead of managing through a table. The data can be filtered and arranged based on completion or spot correlations with a dendrogram or heatmap.

The leather library

Christopher Groskopf, the Creator of Leather, has stated that Leather is the Python charting library for those who need charts now and don't care if they're perfect. This library is designed to work with all data types and create charts as SVGs. This library is relatively new and the charts that can be made using this library are the very basics.

4.7 Matplotlib Library

Pyplot

Pyplot is a Matplotlib module that provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python and the advantage of being free and open source. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. The various plots we can utilize using Pyplot are Line Plot, Histogram, Scatter, 3D Plot, Image, Contour, and Polar.

Create a simple plot

```
import matplotlib.pyplot as plt
# Initializing the data
x=[10, 20, 30, 40]
```



```
y =[20, 25, 35, 55]
# plotting the data
plt.plot(x, y)
plt.show()
```

Adding title

The `title()` method in `matplotlib` module is used to specify the title of the visualization depicted and displays the title using various attributes.

- SYNTAX: `matplotlib.pyplot.title(label, fontdict=None, loc='center', pad=None, **kwargs)`
- CODE: `plt.title("Linear Graph")`

Changing of appearance

We can also change the appearance of the title by using the parameters of this function.

- CODE: `plt.title("Linear Graph", fontsize=25, color="green")`

Adding xlabel and ylabel

In layman's terms, the X label and the Y label are the titles given to X-axis and Y-axis respectively. These can be added to the graph by using the `xlabel()` and `ylabel()` methods.

- SYNTAX: `matplotlib.pyplot.xlabel(xlabel, fontdict=None, labelpad=None, **kwargs)`
`matplotlib.pyplot.ylabel(ylabel, fontdict=None, labelpad=None, **kwargs)`
- CODE: `plt.xlabel('X-Axis')`
`plt.ylabel('Y-Axis')`

Setting limits and tick labels

Matplotlib automatically sets the values and the markers(points) of the X and Y axis, however, it is possible to set the limit and markers manually. `xlim()` and `ylim()` functions are used to set the limits of the X-axis and Y-axis respectively. Similarly, `xticks()` and `yticks()` functions are used to set tick labels.

- CODE: `plt.ylim(0, 80)`
`plt.xticks(x, labels=["one", "two", "three", "four"])`

Adding legends

A legend is an area describing the elements of the graph. In simple terms, it reflects the data displayed in the graph's Y-axis. It generally appears as the box containing a small sample of each color on the graph and a small description of what this data means. The attribute `bbox_to_anchor=(x, y)` of `legend()` function is used to specify the coordinates of the legend, and the attribute `ncol` represents the number of columns that the legend has. Its default value is 1.

- SYNTAX: `matplotlib.pyplot.legend(["name1", "name2"], bbox_to_anchor=(x, y), ncol=1)`
- CODE: `plt.legend(["GFG"])`

Classes in matplotlib

The important classes in `matplotlib` are:

1. Figure class
2. Axes class

Figure class

Consider the figure class as the overall window or page on which everything is drawn. It is a top-level container that contains one or more axes. A figure can be created using the `figure()` method.

- SYNTAX: `class matplotlib.figure.Figure(figsize=None, dpi=None, facecolor=None, edgecolor=None, linewidth=0.0, frameon=None, subplotpars=None, tight_layout=None, constrained_layout=None)`

```

# Python program to show pyplot module
import matplotlib.pyplot as plt
from matplotlib.figure import Figure
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]
# Creating a new figure with width = 7 inches and height = 5 inches with face color as green,
edgecolor as red and the line width of the edge as 7
fig = plt.figure(figsize=(7, 5), facecolor='g', edgecolor='b', linewidth=7)
# Creating a new axes for the figure
ax = fig.add_axes([1, 1, 1, 1])
# Adding the data to be plotted
ax.plot(x, y)
# Adding title to the plot
plt.title("Linear graph", fontsize=25, color="yellow")
# Adding label on the y-axis
plt.ylabel('Y-Axis')
# Adding label on the x-axis
plt.xlabel('X-Axis')
# Setting the limit of y-axis
plt.ylim(0, 80)
# setting the labels of x-axis
plt.xticks(x, labels=["one", "two", "three", "four"])
# Adding legends
plt.legend(['GFG'])
plt.show()

```

Axes Class

Axes class is the most basic and flexible unit for creating sub-plots. A given figure may contain many axes, but a given axes can only be present in one figure. The `axes()` function creates the axes object.

- SYNTAX: `axes([left, bottom, width, height])`

Just like pyplot class, axes class also provides methods for adding titles, legends, limits, labels, etc.

- Adding Title - `ax.set_title()`
- Adding X Label and Y label - `ax.set_xlabel()`, `ax.set_ylabel()`
- Setting Limits - `ax.set_xlim()`, `ax.set_ylim()`
- Tick labels - `ax.set_xticklabels()`, `ax.set_yticklabels()`
- Adding Legends - `ax.legend()`

```

# Python program to show pyplot module
import matplotlib.pyplot as plt
from matplotlib.figure import Figure
# initializing the data
x = [10, 20, 30, 40]

```

```

y = [20, 25, 35, 55]
fig = plt.figure(figsize = (5, 4))
# Adding the axes to the figure
ax = fig.add_axes([1, 1, 1, 1])
# plotting 1st dataset to the figure
ax1 = ax.plot(x, y)
# plotting 2nd dataset to the figure
ax2 = ax.plot(y, x)
# Setting Title
ax.set_title("Linear Graph")
# Setting Label
ax.set_xlabel("X-Axis")
ax.set_ylabel("Y-Axis")
# Adding Legend
ax.legend(labels = ('line 1', 'line 2'))
plt.show()

```

Different types of matplotlib plots

Matplotlib supports a variety of plots including line charts, bar charts, histograms, scatter plots, etc.

Line Chart: It is one of the basic plots and can be created using the `plot()` function. It is used to represent a relationship between two data X and Y on a different axis.

SYNTAX: `matplotlib.pyplot.plot(*args, scalex=True, scaley=True, data=None, **kwargs)`

```

import matplotlib.pyplot as plt
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]
# plotting the data
plt.plot(x, y)
# Adding title to the plot
plt.title("Line Chart")
# Adding label on the y-axis
plt.ylabel("Y-Axis")
# Adding label on the x-axis
plt.xlabel("X-Axis")
plt.show()

```

Properties of Line Chart

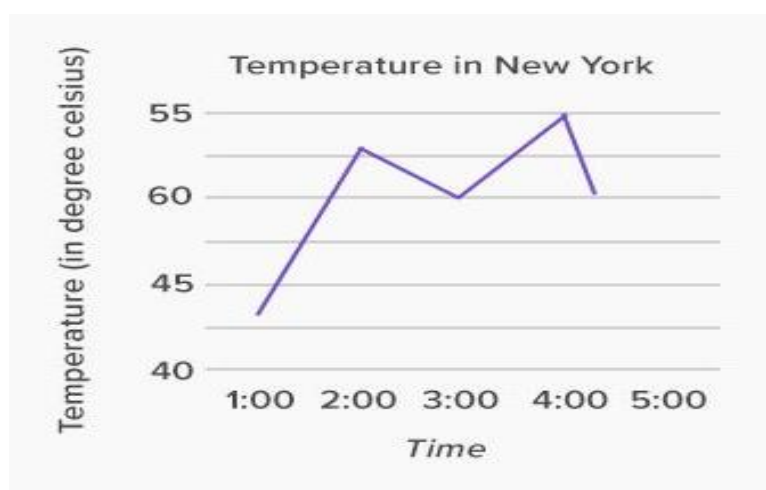
- **color:** Changing the color of the line
- **line width:** Customizing the width of the line
- **marker:** For changing the style of actual plotted point
- **marker size:** For changing the size of the markers
- **line style:** For defining the style of the plotted line

```
import matplotlib.pyplot as plt
# initializing the data
x = [10, 20, 30, 40]
y = [20, 25, 35, 55]
# plotting the data
plt.plot(x, y, color='green', linewidth=3, marker='o', markersize=15, linestyle='--')
# Adding title to the plot
plt.title("Line Chart")
# Adding label on the y-axis
plt.ylabel('Y-Axis')
# Adding label on the x-axis
plt.xlabel('X-Axis')
plt.show()
```

When to use line graph

The line graph is good at:

- 1) Showing specific values
- 2) Trends
- 3) Trends in groups (using multiple line graphs)



Bar Chart

A **bar** chart is a graph that represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent. The bar plots can be plotted horizontally or vertically. A bar chart describes the comparisons between the discrete categories. It can be created using the `bar()` method. We will use the tips dataset. Tips database is the record of the tip given by the customers in a restaurant for two and a half months in the early 1990s. It contains 6 columns as `total_bill`, `tip`, `sex`, `smoker`, `day`, `time`, `size`.

```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# initializing the data
x = data['day']
```

```

y = data['total_bill']
# plotting the data
plt.bar(x, y)
# Adding title to the plot
plt.title("Tips Dataset")
# Adding label on the y-axis
plt.ylabel("Total Bill")
# Adding label on the x-axis
plt.xlabel('Day')
plt.show()

```

Bar Chart Customization

Customization that is available for the Bar Chart -

- **color:** For the bar faces
- **edgecolor:** Color of edges of the bar
- **linewidth:** Width of the bar edges
- **width:** Width of the bar

```

import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# initializing the data
x = data['day']
y = data['total_bill']
# plotting the data
plt.bar(x, y, color='green', edgecolor='blue', linewidth=2)
# Adding title to the plot
plt.title("Tips Dataset")
# Adding label on the y-axis
plt.ylabel("Total Bill")
# Adding label on the x-axis
plt.xlabel('Day')
plt.show()

```

When to use bar chart

Bar charts are best suited when there is a need of comparison of relative point values.

Histograms

A **histogram** is basically used to represent data provided in a form of some groups. It is a type of bar plot where the X-axis represents the bin ranges while the Y-axis gives information about frequency. The `hist()` function is used to compute and create histogram of `x`.

SYNTAX: `matplotlib.pyplot.hist(x, bins=None, range=None, density=False, weights=None, cumulative=False, bottom=None, histtype='bar', align='mid', orientation='vertical', rwidth=None, log=False, color=None, label=None, stacked=False, *, data=None, **kwargs)`

```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# initializing the data
x = data['total_bill']
# plotting the data
plt.hist(x)
# Adding title to the plot
plt.title("Tips Dataset")
# Adding label on the y-axis
plt.ylabel('Frequency')
# Adding label on the x-axis
plt.xlabel('Total Bill')
plt.show()
```

Histogram Customization

Customization that is available for the Histogram -

- **bins:** Number of equal-width bins
- **color:** For changing the face color
- **edgecolor:** Color of the edges
- **linestyle:** For the edgelines
- **alpha:** blending value, between 0 (transparent) and 1 (opaque)

```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# Initializing the data
x = data['total_bill']
# Plotting the data
plt.hist(x, bins=25, color='green', edgecolor='blue', linestyle='--', alpha=0.5)
# Adding title to the plot
plt.title("Tips Dataset")
# Adding label on the y-axis
plt.ylabel('Frequency')
# Adding label on the x-axis
plt.xlabel('Total Bill')
plt.show()
```

Scatter Plots

Scatter plots are used to observe relationships between variables. The `scatter()` method in the `matplotlib` library is used to draw a scatter plot.

SYNTAX: `matplotlib.pyplot.scatter(x_axis_data, y_axis_data, s=None, c=None, marker=None, cmap=None, vmin=None, vmax=None, alpha=None, linewidths=None, edgecolors=None)`

```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# Initializing the data
x = data['day']
y = data['total_bill']
# Plotting the data
plt.scatter(x, y)
# Adding title to the plot
plt.title("Tips Dataset")
# Adding label on the y-axis
plt.ylabel('Total Bill')
# Adding label on the x-axis
plt.xlabel('Day')
plt.show()
```

Scatter plot customization

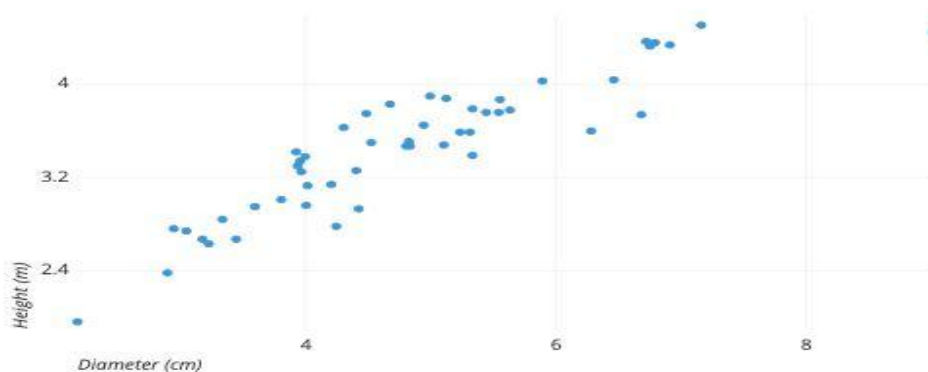
Customizations that are available for the scatter plot are –

- **s:** marker size (can be scalar or array of size equal to size of x or y)
- **c:** color of sequence of colors for markers
- **marker:** marker style
- **linewidths:** width of marker border
- **edgecolor:** marker border color
- **alpha:** blending value, between 0 (transparent) and 1 (opaque)

```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# initializing the data
x = data['day']
y = data['total_bill']
# plotting the data
plt.scatter(x, y, c=data['size'], s=data['total_bill'], marker='D', alpha=0.5)
# Adding title to the plot
plt.title("Tips Dataset")
# Adding label on the y-axis
plt.ylabel('Total Bill')
# Adding label on the x-axis
plt.xlabel('Day')
plt.show()
```

When to use scatter plot?

Scatter plots are needed to convey overall impression of relationship between two variables. It is effective if a relationship exists between the two variables.



Pie chart

A **Pie chart** is a circular chart used to display only one series of data. The area of slices of the pie represents the percentage of the parts of the data. The slices of pie are called wedges. It can be created using the `pie()` method.

SYNTAX: `matplotlib.pyplot.pie(data, explode=None, labels=None, colors=None, autopct=None, shadow=False)`

```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# Initializing the data
cars = ['AUDI', 'BMW', 'FORD', 'TESLA', 'JAGUAR',]
data = [23, 10, 35, 15, 12]
# Plotting the data
plt.pie(data, labels=cars)
# Adding title to the plot
plt.title("Car data")
plt.show()
```

Pie chart customization

Customizations that are available for the Pie chart are

- **explode:** Moving the wedges of the plot
- **autopct:** Label the wedge with their numerical value.
- **color:** Attribute is used to provide color to the wedges.
- **shadow:** Used to create shadow of wedge.

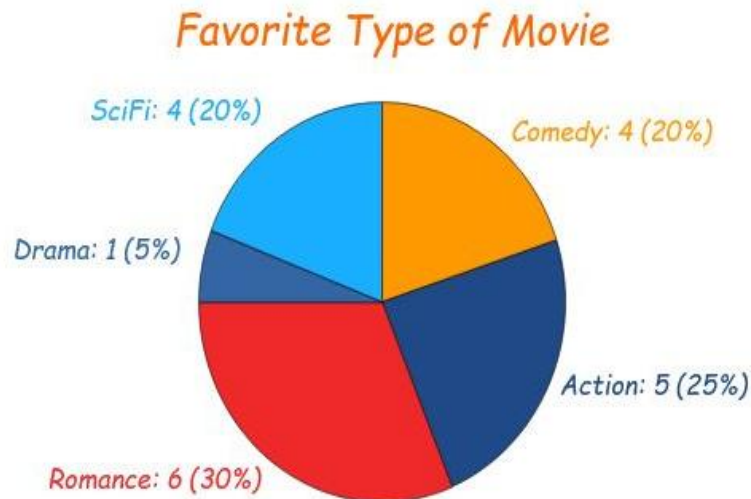
```
import matplotlib.pyplot as plt
import pandas as pd
# Reading the tips.csv file
data = pd.read_csv('tips.csv')
# Initializing the data
```



```
cars = ['AUDI', 'BMW', 'FORD', 'TESLA', 'JAGUAR',]
data = [23, 13, 35, 15, 12]
explode = [0.1, 0.5, 0, 0, 0]
colors = ("orange", "cyan", "yellow", "grey", "green",)
# Plotting the data
plt.pie(data, labels=cars, explode=explode, autopct='%1.2f%%', colors=colors, shadow=True)
plt.show()
```

When to use pie chart

It is for emphasizing differences in proportion among a few numbers



Saving a plot

For saving a plot in a file on storage disk, `savefig()` method is used. A file can be saved in many formats like .png, .jpg, .pdf, etc.

SYNTAX: `pyplot.savefig(fname, dpi=None, facecolor='w', edgecolor='w', orientation='portrait', papertype=None, format=None, transparent=False, bbox_inches=None, pad_inches=0.1, frameon=None, metadata=None)`

```
import matplotlib.pyplot as plt
# Creating data
year = ['2010', '2002', '2004', '2006', '2008']
production = [25, 15, 35, 30, 10]
# Plotting barchart
plt.bar(year, production)
# Saving the figure.
plt.savefig("output.jpg")
```

4.8 Advanced Data Visualization using Seaborn Library

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It is built on the top of matplotlib library and closely integrated into the data structures from pandas. **Pandas** offer tools for cleaning and process your data. It is the most popular Python library that is used for data analysis. In pandas, a data table is called a data frame. Sea born helps to visualize the statistical relationships. To understand how variables in a dataset are related to one another and how that relationship is dependent on other variables, we perform statistical analysis. This Statistical analysis helps to visualize the trends and identify various patterns in the dataset.

```

Creating Pandas dataframe
# Python code demonstrate creating
import pandas as pd
# Initialise data of lists.
data = {'Name':['Mohe' , 'Karnal' , 'Yrik' , 'jack' ],'Age':[ 30 , 21 , 29 , 28 ]}
# Create DataFrame
df = pd.DataFrame( data )
# Print the output.
df

```

4.9 Visualization Libraries

- Line Plot
- Scatter Plot
- Box plot
- Point plot
- Count plot
- Violin plot
- Swarm plot
- Bar plot
- KDE Plot

Line Plot

Line plot Is the most popular plot to draw a relationship between x and y with the possibility of several semantic groupings.

Syntax: `sns.lineplot(x=None, y=None)`

Parameters: x, y: Input data variables; must be numeric. Can pass data directly or reference columns in data.

```

# import module
import seaborn as sns
import pandas
# loading csv
data = pandas.read_csv("nba.csv")
# plotting lineplot
sns.lineplot( data['Age'], data['Weight'])

```

Hue Parameter

```

# import module
import seaborn as sns
import pandas
# read the csv data
data = pandas.read_csv("nba.csv")
# plot
sns.lineplot(data['Age'],data['Weight'], hue =data["Position"])

```

Scatter Plot

Scatterplot Can be used with several semantic groupings which can help to understand well in a graph against continuous/categorical data. It can draw a two-dimensional graph.

Syntax: `seaborn.scatterplot(x=None, y=None)`

Parameters: x, y: Input data variables that should be numeric.

Returns: This method returns the Axes object with the plot drawn onto it.

```
# import module
import seaborn
import pandas
# load csv
data = pandas.read_csv("nba.csv")
# plotting
seaborn.scatterplot(data['Age'],data['Weight'])
```

Hue Parameter

```
import seaborn
import pandas
data = pandas.read_csv("nba.csv")
seaborn.scatterplot( data['Age'], data['Weight'], hue =data["Position"])
```

Box Plot

A box plot (or box-and-whisker plot) is the visual representation of the depicting groups of numerical data through their quartiles against continuous/categorical data. A box plot consists of 5 things.

- Minimum
- First Quartile or 25%
- Median (Second Quartile) or 50%
- Third Quartile or 75%
- Maximum

Syntax: `seaborn.boxplot(x=None, y=None, hue=None, data=None)`

Parameters: x, y, hue: Inputs for plotting long-form data.

Data: Dataset for plotting. If x and y are absent, this is interpreted as wide form.

Returns: It returns the Axes object with the plot drawn onto it.

```
# import module
import seaborn as sns
import pandas
# read csv and plotting
data = pandas.read_csv( "nba.csv" )
sns.boxplot( data['Age'] )
```

Violin Plot

A violin plot is like a boxplot. It shows several quantitative data across one or more categorical variables such that those distributions can be compared.

Syntax: `seaborn.violinplot(x=None, y=None, hue=None, data=None)`

Parameters: x, y, hue: Inputs for plotting long-form data.

Data: Dataset for plotting.

```
# import module
import seaborn as sns
import pandas
# read csv and plot
data = pandas.read_csv("nba.csv")
sns.violinplot(data['Age'])
```

Swarm plot

A swarm plot is similar to a strip plot, we can draw a swarm plot with non-overlapping points against categorical data.

Syntax: `seaborn.swarmplot(x=None, y=None, hue=None, data=None)`

Parameters: x, y, hue: Inputs for plotting long-form data.

Data: Dataset for plotting.

```
# import module
import seaborn
seaborn.set(style = 'whitegrid')
# read csv and plot
data = pandas.read_csv( "nba.csv" )
seaborn.swarmplot(x = data["Age"])
```

Bar Plot

Bar plot represents an estimate of central tendency for a numeric variable with the height of each rectangle and provides some indication of the uncertainty around that estimate using error bars.

Syntax: `seaborn.barplot(x=None, y=None, hue=None, data=None)`

Parameters: x, y: This parameter take names of variables in data or vector data, Inputs for plotting long-form data.

Hue: (optional) This parameter takes column name for color encoding.

Data: (optional) This parameter take DataFrame, array, or list of arrays, Dataset for plotting. If x and y are absent, this is interpreted as wide form. Otherwise, it is expected to be long form.

Returns: Returns the Axes object with the plot drawn onto it.

```
# import module
import seaborn
seaborn.set(style = 'whitegrid')
# read csv and plot
data = pandas.read_csv("nba.csv")
seaborn.barplot(x =data["Age"])
```

Point plot

Point plot used to show point estimates and confidence intervals using scatter plot glyphs. A point plot represents an estimate of central tendency for a numeric variable by the position of scatter plot points and provides some indication of the uncertainty around that estimate using error bars.

Syntax: `seaborn.pointplot(x=None, y=None, hue=None, data=None)`

Parameters: x, y: Inputs for plotting long-form data.

Hue: (optional) column name for color encoding.

Data: dataframe as a Dataset for plotting.

Return: The Axes object with the plot drawn onto it.

```
# import module
import seaborn
seaborn.set(style = 'whitegrid')
# read csv and plot
data = pandas.read_csv("nba.csv")
seaborn.pointplot(x = "Age", y = "Weight", data = data)
```

Count Plot

Count plot used to Show the counts of observations in each categorical bin using bars.

Syntax: `seaborn.countplot(x=None, y=None, hue=None, data=None)`

Parameters: `x, y`: This parameter take names of variables in data or vector data, optional, Inputs for plotting long-form data.

Hue: (optional) This parameter takes column name for color encoding.

Data: (optional) This parameter take DataFrame, array, or list of arrays, Dataset for plotting. If `x` and `y` are absent, this is interpreted as wide form. Otherwise, it is expected to be long form.

Returns: Returns the Axes object with the plot drawn onto it.

```
# import module
import seaborn
seaborn.set(style = 'whitegrid')
# read csv and plot
data = pandas.read_csv("nba.csv")
seaborn.countplot(data["Age"])
```

KDE Plot

KDE Plot described as **Kernel Density Estimate** is used for visualizing the Probability Density of a continuous variable. It depicts the probability density at different values in a continuous variable. We can also plot a single graph for multiple samples which helps in more efficient data visualization.

Syntax: `seaborn.kdeplot(x=None, *, y=None, vertical=False, palette=None, **kwargs)`

Parameters: `x, y`: vectors or keys in data

Vertical: boolean (True or False)

Data: `pandas.DataFrame, numpy.ndarray, mapping, or sequence`

```
# import module
import seaborn as sns
import pandas
# read top 5 column
data = pandas.read_csv("nba.csv").head()
sns.kdeplot( data['Age'], data['Number'])
```

Summary

- Visualization is the graphical representation of data that can make information easy to analyze and understand.

- The data visualization software applications differ in their ability to use various types of graphs and visuals, simplified user interface, accurate trend tracking capability, level of security, ease of use in mobile and friendly report generation.
- Zoho Analytics uses a variety of tools, such as pivot tables, KPI widgets, and tabular view components. It can generate reports with valuable business insights.
- Microsoft Power BI comes with unlimited access to on-site and in-cloud data that gives a centralized data access hub.
- The matplotlib library was created by John D. Hunter and is currently maintained by a team of Python developers.
- The matplotlib allows easy use of labels, axes titles, grids, legends, and other graphic requirements with customizable values and text.
- The seaborn library follows creative styles and rich color palettes, that allows to create visualization plots to be more attractive and modern.
- It is integrated with pandas and is mainly used for creating very simple graphics. This library sacrifices the complexity of plotting complex graphs as its primary focus is on building basic graphs that are often required for analyzing simple data distribution.
- One interesting fact about plotly is that the graphs are not saved as images but rather serialized as JSON, because of which the graphs can be opened and viewed with other applications such as R, Julia, and MATLAB.

Keywords

- **Data visualization:** It is the graphical representation of data that can make information easy to analyze and understand.
- **Visual encoding:** It is the approach used to map data into visual structures, thereby building an image on the screen.
- **Qlikview:** It allows users to create default and custom data connectors and templates, depending on one's need.
- **Sisense:** It uses agile analysis software with a variety of data visualization options. It can create dashboards and graphics with drag and drop user interface.
- **Seaborn library:** Seaborn library is considered to be a higher-level library, there are certain special visualization tools such as, violin plots, heat maps and time series plots that can be created using this library.
- **Ggplot library:** The ggplot library is based on the ggplot2 library which is an R plotting system and concepts are based on the Grammar of Graphics.
- **Bokeh library:** The Bokeh library is native to Python and is mainly used to create interactive, web-ready plots, which can be easily output as HTML documents, JSON objects, or interactive web applications.
- **Plotly library:** The plotly library is an online platform for data visualization and it can be used in making interactive plots that are not possible using other Python libraries.
- **Pygal library:** The pygal library creates interactive plots that can be embedded in the web browser.
- **Geoplotlib library:** The geoplotlib is a toolbox for designing maps and plotting geographical data. Few of the map-types that can be created are heatmaps, dotdensity maps, and choropleths.

SelfAssessment

1. How can we illustrate the complex data relationship and patterns?
 - A. Lines
 - B. Graphs
 - C. Colours
 - D. All of the above

2. Data visualization can help in
 - A. Identifying outliers
 - B. Improve response time
 - C. Easier visualization of patterns
 - D. All of the above

3. Which approach used to map data into visual structures, thereby building an image on the screen?
 - A. Visual Encoding
 - B. Audio Encoding
 - C. Redirecting
 - D. None of the above

4. What is the visualization graph supposed to display?
 - A. Comparison
 - B. Composition
 - C. Connection
 - D. All of the above

5. Which of the following can be used to show the composition?
 - A. Pie chart
 - B. Waterfall chart
 - C. Stacked column chart
 - D. All of the above

6. The Tableau data visualization software is
 - A. A flexible data analysis method
 - B. Mobile friendly
 - C. Manageable
 - D. All of the above

7. The plotly supports
 - A. 2D and 3D chart options
 - B. Designer input
 - C. Open source coding
 - D. All of the above mentioned

8. Which is the first data visualization library developed in Python?
- A. Matplotlib library
 - B. Seaborn library
 - C. Bokeh library
 - D. None of the above
9. Which library supports the three varying interfaces used by different types of users?
- A. Matplotlib library
 - B. Seaborn library
 - C. Bokeh library
 - D. None of the above
10. What is a Matplotlib module that provides a MATLAB-like interface?
- A. Aypplot
 - B. Pyplot
 - C. Syplot
 - D. Typlot
11. Which of these are the important classes of matplotlib library?
- A. Axes class
 - B. Figure class
 - C. Both of the above
 - D. None of the above
12. How many components are required to make a box plot?
- A. 4
 - B. 5
 - C. 6
 - D. 7
13. Which of the following is used to show the comparison of relative data points?
- A. Line graph
 - B. Scatter graph
 - C. Bar graph
 - D. Pie graph
14. Which of these is most effective if a relationship exists between the two variables
- A. Line graph
 - B. Scatter plot
 - C. Bar graph
 - D. Pie plot

15. Which of the following emphasizes differences in proportion among a few numbers
- A. Line graph
 - B. Scatter plot
 - C. Bar graph
 - D. Pie plot

Answers for Self Assessment

1. D 2. D 3. A 4. D 5. D
6. D 7. D 8. A 9. C 10. C
11. C 12. B 13. C 14. B 15. D

Review Questions

1. What is data visualization? Explain its need and importance.
2. Explain the need of data visualization for different purposes. Also explain its advantages.
3. What is visual encoding? Also explain few retinal variables.
4. Explain the role of data visualization in different areas and tell the corresponding data visualization graph.
5. Describe few data visualization softwares. Also tell its important key features.
6. Name few important data visualization libraries. Also explain in detail about matplotlib library.
7. What is Pyplot module? How can we create a simple plot using it? Also tell how to add different things to the plot?
8. Which type of plots can be created using matplotlib library? Also explain its syntax.
9. What is advanced data visualization? How can we do it using seaborn library?



Further Readings

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>



Web Links

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Unit 05: Role of Statistics in Data Science

CONTENTS

Objectives

Introduction

5.1 Key Features in Hypothesis Testing

5.2 Null and Alternative Hypothesis

5.3 Type 1 and Type 2 Errors

5.4 P-Value/ Probability Value

5.5 ANOVA

5.6 Chi-Square Test

Summary

Keywords

Self Assessment

Answer for Self Assessment

Review Questions

Further Readings

Objectives

After studying this unit, you will be able to

- understand what is hypothesis testing,
- understand the steps of hypothesis testing,
- understand two types of hypotheses,
- understand Type - I and Type -II errors,
- understand what P-value is,
- understand ANOVA,
- understand chi-square test.

Introduction

Hypothesis testing is an act in statistics whereby an analyst tests an assumption regarding a population parameter. The methodology employed by the analyst depends on the nature of the data used and the reason for the analysis. Hypothesis testing is used to assess the plausibility of a hypothesis by using sample data. Such data may come from a larger population, or from a data-generating process.

5.1 Key Features in Hypothesis Testing

- Hypothesis testing is used to assess the plausibility of a hypothesis by using sample data.
- The test provides evidence concerning the plausibility of the hypothesis, given the data.
- Statistical analysts test a hypothesis by measuring and examining a random sample of the population being analyzed.

Statistical analysts test a hypothesis by measuring and examining a random sample of the population being analyzed. All analysts use a random population sample to test two different

hypotheses: the null hypothesis and the alternative hypothesis. In hypothesis testing, an analyst tests a statistical sample, with the goal of providing evidence on the plausibility of the null hypothesis.

5.2 Null and Alternative Hypothesis

- **Null Hypothesis:** The null hypothesis is usually a hypothesis of equality between population parameters; e.g., a null hypothesis may state that the population mean return is equal to zero.
- **Alternate Hypothesis:** The alternative hypothesis is effectively the opposite of a null hypothesis (e.g., the population mean return is not equal to zero).

The null and alternative hypothesis are mutually exclusive, and only one can be true. However, one of the two hypotheses will always be true.

Steps of hypothesis testing

1. The first step is for the analyst to state the two hypotheses so that only one can be right.
2. The second step is to collect data.
3. The third step is to perform a statistical test.
4. The fourth is to reject the null hypothesis, or state that the null hypothesis is plausible, given the data.
5. The fifth step is to present your findings.

Step 1: Null and alternate hypothesis

The alternate hypothesis is usually your initial hypothesis that predicts a relationship between variables. The null hypothesis is a prediction of no relationship between the variables you are interested in.

Step 2: Collect data

For a statistical test to be valid, it is important to perform sampling and collect data in a way that is designed to test your hypothesis. If your data are not representative, then you cannot make statistical inferences about the population you are interested in.

Step 3: Perform a statistical test

There are a variety of statistical tests available, but they are all based on the comparison of within-group variance (how spread out the data is within a category) versus between-group variance (how different the categories are from one another).

Step 4: Decide whether to reject or fail to reject your null hypothesis

Based on the outcome of your statistical test, you will have to decide whether to reject or fail to reject your null hypothesis. In most cases you will use the p-value generated by your statistical test to guide your decision.

Step 5: Present your findings

Present your findings by either any article or by the means of a research publications of by any form of IPR.



Example: You want to test whether there is a relationship between gender and height. Based on your knowledge of human physiology, you formulate a hypothesis that men are,

on average, taller than women.

Step 1: State null hypothesis and alternative hypothesis.

Ho: Men are, on average, not taller than women.

Ha: Men are, on average, taller than women.

Step 2: Collection of data. Your sample should have an equal proportion of men and women and cover a variety of socio-economic classes and any other control variables that might influence average height.

Step 3: Perform a statistical test

Step 4: Decide whether to reject or fail to reject your null hypothesis. In your analysis of the difference in average height between men and women, you find that the p-value of 0.002 is below your cutoff of 0.05, so you decide to reject your null hypothesis of no difference.

Step 5: Present your findings.



Example: A person wants to test that a penny has exactly a 50% chance of landing on heads, the null hypothesis would be that 50% is correct, and the alternative hypothesis would be that 50% is not correct.

Mathematically, the null hypothesis would be represented as $H_0: P = 0.5$. The alternative hypothesis would be denoted as " H_a " and be identical to the null hypothesis, except with the equal sign struck-through, meaning that it does not equal 50%. A random sample of 100-coin flips is taken, and the null hypothesis is then tested.

If it is found that the 100-coin flips were distributed as 40 heads and 60 tails, the analyst would assume that a penny does not have a 50% chance of landing on heads and would reject the null hypothesis and accept the alternative hypothesis.

If, on the other hand, there were 48 heads and 52 tails, then it is plausible that the coin could be fair and still produce such a result.

In cases such as this where the null hypothesis is "accepted," the analyst states that the difference between the expected results (50 heads and 50 tails) and the observed results (48 heads and 52 tails) is "explainable by chance alone."

5.3 Type 1 and Type 2 Errors

In statistics, a Type I error is a false positive conclusion, while a Type II error is a false negative conclusion. You decide to get tested for COVID-19 based on mild symptoms. There are two errors that could potentially occur:

- **Type I error (false positive):** the test result says you have coronavirus, but you actually don't.
- **Type II error (false negative):** the test result says you don't have coronavirus, but you actually do.

Making a statistical decision always involves uncertainties, so the risks of making these errors are unavoidable in hypothesis testing.

Alpha and Beta

The probability of making a Type I error is the significance level, or alpha (α), while the probability of making a Type II error is beta (β). These risks can be minimized through careful planning in your study design.

Using hypothesis testing, you can make decisions about whether your data support or refute your research predictions. Hypothesis testing starts with the assumption of no difference between groups or no relationship between variables in the population—this is the **null hypothesis**. It's always paired with an **alternative hypothesis**, which is your research prediction of an actual difference between groups or a true relationship between variables.



Example: To test whether a new drug intervention can alleviate symptoms of an autoimmune disease.

Null hypothesis (H_0) is that the new drug has no effect on symptoms of the disease.

Alternative hypothesis (H_1) is that the drug is effective for alleviating symptoms of the disease.

A **Type I error** happens when you get false positive results: you conclude that the drug intervention improved symptoms when it didn't. These improvements could have arisen from other random factors or measurement errors.

A **Type II error** happens when you get false negative results: you conclude that the drug intervention didn't improve symptoms when it did. Your study may have missed key indicators of improvements or attributed any improvements to other factors instead.

Type I and Type II Error

Null hypothesis is ...	True	False
Rejected	Type I error False positive Probability = α	Correct decision True positive Probability = $1 - \beta$
Not rejected	Correct decision True negative Probability = $1 - \alpha$	Type II error False negative Probability = β

Type 1 error and its statistical significance

A Type I error means rejecting the null hypothesis when it's true. It means concluding that results are **statistically significant** when they came about purely by chance or because of unrelated factors. The risk of committing this error is the significance level (alpha or α) you choose. That's a value that you set at the beginning of your study to assess the statistical probability of obtaining your results. The significance level is usually set at 0.05 or 5%. This means that your results only have a 5% chance of occurring, or less, if the null hypothesis is true.

In your clinical study, you compare the symptoms of patients who received the new drug intervention or a control treatment. Using a t test, you obtain a p value of .035. This p value is lower than your alpha of .05, so you consider your results statistically significant and reject the null hypothesis. However, the p value means that there is a 3.5% chance of your results occurring if the null hypothesis is true. Therefore, there is still a risk of making a Type I error. Power is the extent to which a test can correctly detect a real effect when there is one. A power level of 80% or higher is usually considered acceptable.

Type 2 error and its statistical significance

A Type II error means not rejecting the null hypothesis when it's actually false. This is not quite the same as "accepting" the null hypothesis, because hypothesis testing can only tell you whether to reject the null hypothesis. Instead, a Type II error means failing to conclude there was an effect when there actually was. In reality, your study may not have had enough **statistical power** to detect an effect of a certain size. When preparing your clinical study, you complete a power analysis and determine that with your sample size, you have an 80% chance of detecting an effect size of 20% or greater. An effect size of 20% means that the drug intervention reduces symptoms by 20% more than the control treatment. However, a Type II may occur if an effect that's smaller than this size. A smaller effect size is unlikely to be detected in your study due to inadequate statistical power.

Statistical power is determined by:

- Size of the effect: Larger effects are more easily detected.
- Measurement error: Systematic and random errors in recorded data reduce power.
- Sample size: Larger samples reduce sampling error and increase power.
- Significance level: Increasing the significance level increases power.

Trade-off between Type I and Type II errors

The Type I and Type II error rates influence each other. That's because the significance level (the Type I error rate) affects statistical power, which is inversely related to the Type II error rate. This means there's an important tradeoff between Type I and Type II errors:

- Setting a lower significance level decreases a Type I error risk but increases a Type II error risk.
- Increasing the power of a test decreases a Type II error risk but increases a Type I error risk.

5.4 P-Value/Probability Value

The p -value is a number, calculated from a statistical test, that describes how likely you are to have found a particular set of observations if the null hypothesis were true. P -values are used in hypothesis testing to help decide whether to reject the null hypothesis. The smaller the p -value, the more likely you are to reject the null hypothesis.

All statistical tests have a null hypothesis. For most tests, the null hypothesis is that there is no relationship between your variables of interest or that there is no difference among groups. For example, in a two-tailed t -test, the null hypothesis is that the difference between two groups is zero.

- You want to know whether there is a difference in longevity between two groups of mice fed on different diets, diet A and diet B. You can statistically test the difference between these two diets using a two-tailed t test.
- Null hypothesis: there is no difference in longevity between the two groups.
- Alternative hypothesis: there is a difference in longevity between the two groups.

The p -value tells you how likely it is that your data could have occurred under the null hypothesis. It does this by calculating the likelihood of your test statistic, which is the number calculated by a statistical test using your data. The p -value tells you how often you would expect to see a test statistic as extreme or more extreme than the one calculated by your statistical test if the null hypothesis of that test was true. The p -value gets smaller as the test statistic calculated from your data gets further away from the range of test statistics predicted by the null hypothesis. The p -value is a proportion: if your p -value is 0.05, that means that 5% of the time you would see a test statistic at least as extreme as the one you found if the null hypothesis was true.

Calculation of p -value

P -values are usually automatically calculated by your statistical program (R, SPSS, etc.). You can also find tables for estimating the p -value of your test statistic online. These tables show, based on the test statistic and degrees of freedom (number of observations minus number of independent variables) of your test, how frequently you would expect to see that test statistic under the null hypothesis. The calculation of the p -value depends on the statistical test you are using to test your hypothesis. Different statistical tests have different assumptions and generate different test statistics. You should choose the statistical test that best fits your data and matches the effect or relationship you want to test. The number of independent variables you include in your test changes how large or small the test statistic needs to be to generate the same p -value.

Choosing a statistical test

If you are comparing only two different diets, then a two-sample t-test is a good way to compare the groups. To compare three different diets, use an ANOVA instead – doing multiple pairwise comparisons will result in artificially low p-values and lets you overestimate the significance of the difference between groups.

P-values are most often used by researchers to say whether a certain pattern they have measured is statistically significant. Statistical significance is another way of saying that the p-value of a statistical test is small enough to reject the null hypothesis of the test.

5.5 ANOVA

Analysis of variance (ANOVA) is a collection of statistical models and their associated estimation procedures (such as the "variation" among and between groups) used to analyze the differences among means. ANOVA was developed by the statistician Ronald Fisher. ANOVA is based on the law of total variance, where the observed variance in a particular variable is partitioned into components attributable to different sources of variation. In its simplest form, ANOVA provides a statistical test of whether two or more population means are equal, and therefore generalizes the t-test beyond two means. In other words, the ANOVA is used to test the difference between two or more means. Due to its nature, it can only test general differences only. A non-parametric alternative is PERMANOVA.

Classes of models

There are three classes of models used in analysis of variance:

- 1) Fixed effects models
- 2) Random effects models
- 3) Mixed effects models

Fixed-effects models: The fixed-effects model (class I) of analysis of variance applies to situations in which the experimenter applies one or more treatments to the subjects of the experiment to see whether the response variable values change. This allows the experimenter to estimate the ranges of response variable values that the treatment would generate in the population.

Random-effects model: Random-effects model (class II) is used when the treatments are not fixed. This occurs when the various factor levels are sampled from a larger population. Because the levels themselves are random variables, some assumptions, and the method of contrasting the treatments (a multi-variable generalization of simple differences) differ from the fixed-effects model.

Mixed-effects model: A mixed-effects model (class III) contains experimental factors of both fixed and random-effect types, with appropriately different interpretations and analysis for the two types.

ANOVA Test

An ANOVA test is a way to find out if survey or experiment results are significant. In other words, they help you to figure out if you need to reject the null hypothesis or accept the alternate hypothesis. Basically, you're testing groups to see if there's a difference between them. Examples of when you might want to test different groups:

- 1) A group of psychiatric patients are trying three different therapies: counseling, medication and biofeedback. You want to see if one therapy is better than the others.
- 2) A manufacturer has two different processes to make light bulbs. They want to know if one process is better than the other.
- 3) Students from different colleges take the same exam. You want to see if one college outperforms the other.

What does “one-way” or “two-way” mean?

One-way or two-way refers to the number of independent variables (IVs) in your Analysis of Variance test. One-way has one independent variable (with 2 levels). For example: brand of cereal, two-way has two independent variables (it can have multiple levels). For example: brand of cereal, calories.

“Levels” or “groups”

Groups or levels are different groups within the same independent variable. Your levels for “brand of cereal” might be Lucky Charms, Raisin Bran, Cornflakes – a total of three levels. Your levels for “Calories” might be sweetened, unsweetened – a total of two levels. Let’s say you are studying if an alcoholic support group and individual counseling combined is the most effective treatment for lowering alcohol consumption. You might split the study participants into three groups or levels:

- Medication only,
- Medication and counseling,
- Counseling only.

Your dependent variable would be the number of alcoholic beverages consumed per day. If your groups or levels have a hierarchical structure (each level has unique subgroups), then use a nested ANOVA for the analysis.

Replication

It’s whether you are replicating (i.e., duplicating) your test(s) with multiple groups. With a two way ANOVA with replication, you have two groups and individuals within that group are doing more than one thing (i.e., two groups of students from two colleges taking two tests). If you only have one group taking two tests, you will use without replication.

Types of Tests

There are two main types: one-way and two-way. Two-way tests can be with or without replication.

One-way ANOVA between groups: It is used when you want to test two groups to see if there’s a difference between them.

Two-way ANOVA without replication: It is used when you have one group and you’re double testing that same group. For example, you’re testing one set of individuals before and after they take a medication to see if it works or not.

Two-way ANOVA with replication: Two groups, and the members of those groups are doing more than one thing. For example, two groups of patients from different hospitals trying two different therapies.

One way ANOVA

A one-way ANOVA is used to compare two means from two independent (unrelated) groups using the F-distribution. The null hypothesis for the test is that the two means are equal. Therefore, a significant result means that the two means are unequal.

When to use One Way ANOVA?

Situation 1: You have a group of individuals randomly split into smaller groups and completing different tasks. For example, you might be studying the effects of tea on weight loss and form three groups: green tea, black tea, and no tea.

Situation 2: Like situation 1, but in this case the individuals are split into groups based on an attribute they possess. For example, you might be studying leg strength of people according to weight.

You could split participants into weight categories (obese, overweight, and normal) and measure their leg strength on a weight machine.

Limitations of One-way ANOVA

A one-way ANOVA will tell you that at least two groups were different from each other. But it won't tell you which groups were different. If your test returns a significant f-statistic, you may need to run an ad hoc test (like the Least Significant Difference test) to tell you exactly which groups had a difference in means.

Two Way ANOVA

A Two Way ANOVA is an extension of the One Way ANOVA. With a One Way, you have one independent variable affecting a dependent variable. With a Two Way ANOVA, there are two independents. Use a two-way ANOVA when you have one measurement variable (i.e., a quantitative variable) and two nominal variables. In other words, if your experiment has a quantitative outcome and you have two categorical explanatory variables, a two-way ANOVA is appropriate. For example, you might want to find out if there is an interaction between income and gender for anxiety level at job interviews. The anxiety level is the outcome, or the variable that can be measured. Gender and Income are the two categorical variables. These categorical variables are also the independent variables, which are called factors in a Two Way ANOVA. The factors can be split into levels. In the above example, income level could be split into three levels: low, middle and high income. Gender could be split into three levels: male, female, and transgender. Treatment groups are all possible combinations of the factors. In this example there would be $3 \times 3 = 9$ treatment groups.

The results from a Two Way ANOVA will calculate a main effect and an interaction effect. The main effect is like a One Way ANOVA: each factor's effect is considered separately. With the interaction effect, all factors are considered at the same time. Interaction effects between factors are easier to test if there is more than one observation in each cell. Two null hypotheses are tested if you are placing one observation in each cell. For this example, those hypotheses would be: H01: All the income groups have equal mean stress.

H02: All the gender groups have equal mean stress.

For multiple observations in cells, you would also be testing a third hypothesis: H03: The factors are independent, or the interaction effect does not exist.

Assumptions for Two-way ANOVA

1. The population must be close to a normal distribution.
2. Samples must be independent.
3. Population variances must be equal (i.e., homoscedastic).
4. Groups must have equal sample sizes.

MANOVA

MANOVA is just an ANOVA with several dependent variables. It's similar to many other tests and experiments in that its purpose is to find out if the response variable (i.e., your dependent variable) is changed by manipulating the independent variable. The test helps to answer many research questions, including:

- Do changes to the independent variables have statistically significant effects on dependent variables?
- What are the interactions among dependent variables?
- What are the interactions among independent variables?

Suppose you wanted to find out if a difference in textbooks affected students' scores in math and science. Improvements in math and science means that there are two dependent variables, so a MANOVA is appropriate. An ANOVA will give you a single (univariate) f-value while a MANOVA will give you a multivariate F value. MANOVA tests the multiple dependent variables by creating new, artificial, dependent variables that maximize group

differences. These new dependent variables are linear combinations of the measured dependent variables.

Advantages and Disadvantages of MANOVA vs ANOVA

Advantages

1. MANOVA enables you to test multiple dependent variables.
2. MANOVA can protect against Type I errors.

Disadvantages

1. MANOVA is many times more complicated than ANOVA, making it a challenge to see which independent variables are affecting dependent variables.
2. One degree of freedom is lost with the addition of each new variable.
3. The dependent variables should be uncorrelated as much as possible.

Factorial ANOVA

A factorial ANOVA is an Analysis of Variance test with more than one independent variable, or “factor”. It can also refer to more than one Level of Independent Variable. For example, an experiment with a treatment group and a control group has one factor (the treatment) but two levels (the treatment and the control). The terms “two-way” and “three-way” refer to the number of factors or the number of levels in your test. Four-way ANOVA and above are rarely used because the results of the test are complex and difficult to interpret. A two-way ANOVA has two factors (independent variables) and one dependent variable. For example, time spent studying and prior knowledge are factors that affect how well you do on a test. A three-way ANOVA has three factors (independent variables) and one dependent variable. For example, time spent studying, prior knowledge, and hours of sleep are factors that affect how well you do on a test. Factorial ANOVA is an efficient way of conducting a test. Instead of performing a series of experiments where you test one independent variable against one dependent variable, you can test all independent variables at the same time.

Variability

In a one-way ANOVA, variability is due to the differences between groups and the differences within groups. In factorial ANOVA, each level and factor are paired up with each other (“crossed”). This helps you to see what interactions are going on between the levels and factors. If there is an interaction, then the differences in one factor depend on the differences in another.

Assumptions of factorial ANOVA

- Normality: the dependent variable is normally distributed.
- Independence: Observations and groups are independent from each other.
- Equality of Variance: the population variances are equal across factors/levels.

Steps

1. Find the mean for each of the groups.
2. Find the overall mean (the mean of the groups combined).
3. Find the Within Group Variation; the total deviation of each member’s score from the Group Mean.
4. Find the Between Group Variation: the deviation of each Group Mean from the Overall Mean.
5. Find the F statistic: the ratio of Between Group Variation to Within Group Variation.

ANOVA vs T Test

- A Student's t-test will tell you if there is a significant variation between groups.
- A t-test compares means, while the ANOVA compares variances between populations.
- You could technically perform a series of t-tests on your data.
- However, as the groups grow in number, you may end up with a lot of pair comparisons that you need to run. ANOVA will give you a single number (the f-statistic) and one p-value to help you support or reject the null hypothesis.

5.6 Chi-Square Test

A chi-square (χ^2) statistic is a test that measures how a model compares to actual observed data. The data used in calculating a chi-square statistic must be random, raw, mutually exclusive, drawn from independent variables, and drawn from a large enough sample. For example, the results of tossing a fair coin meet these criteria. Chi-square tests are often used in hypothesis testing. The chi-square statistic compares the size of any discrepancies between the expected results and the actual results, given the size of the sample and the number of variables in the relationship. For these tests, degrees of freedom are utilized to determine if a certain null hypothesis can be rejected based on the total number of variables and samples within the experiment. As with any statistic, the larger the sample size, the more reliable the results.

Formula for Chi-Square is

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

where:

C = Degrees of freedom

O = Observed value(s)

E = Expected value(s)

What does a Chi-Square Statistic Tell You?

There are two main kinds of chi-square tests: the test of independence, which asks a question of relationship, such as, "Is there a relationship between student sex and course choice? The goodness-of-fit test, which asks something like "How well does the coin in my hand match a theoretically fair coin?"

Independence

When considering student type and course choice, a χ^2 test for independence could be used. To do this test, the researcher would collect data on the two chosen variables (type and courses picked) and then compare the frequencies at which male and female students select among the offered classes. If there is no relationship between type and course selection (that is, if they are independent), then the actual frequencies at which male and female students select each offered course should be expected to be approximately equal, or conversely, the proportion of male and female students in any selected course should be approximately equal to the proportion of male and female students in the sample. A χ^2 test for independence can tell us how likely it is that random chance can explain any observed difference between the actual frequencies in the data and these theoretical expectations.

Goodness of fit

χ^2 provides a way to test how well a sample of data matches the (known or assumed) characteristics of the larger population that the sample is intended to represent. This is known as goodness of fit. If the sample data do not fit the expected properties of the population that we are interested in, then we would not want to use this sample to draw conclusions about the larger population.



Example:

Consider an imaginary coin with exactly a 50/50 chance of landing heads or tails and a real coin that you toss 100 times. If this coin is fair, then it will also have an equal probability of landing on either side, and the expected result of tossing the coin 100 times is that heads will come up 50 times and tails will come up 50 times. In this case, χ^2 can tell us how well the actual results of 100-coin flips compare to the theoretical model that a fair coin will give 50/50 results. The actual toss could come up 50/50, or 60/40, or even 90/10. The farther away the actual results of the 100 tosses is from 50/50, the less good the fit of this set of tosses is to the theoretical expectation of 50/50, and the more likely we might conclude that this coin is not actually a fair coin.

When to use a chi-square test?

A chi-square test is used to help determine if observed results are in line with expected results, and to rule out that observations are due to chance. A chi-square test is appropriate for this when the data being analyzed is from a random sample, and when the variable in question is a categorical variable. A categorical variable is one that consists of selections such as type of car, race, educational attainment, male vs. female, how much somebody likes a political candidate (from very much to very little), etc. These types of data are often collected via survey responses or questionnaires. Therefore, chi-square analysis is often most useful in analyzing this type of data.

What is a chi-square test used for?

Chi-square is a statistical test used to examine the differences between categorical variables from a random sample in order to judge goodness of fit between expected and observed results. Is chi-square analysis used when the independent variable is nominal or ordinal? A nominal variable is a categorical variable that differs by quality, but whose numerical order could be irrelevant. For instance, asking somebody their favorite color would produce a nominal variable. Asking somebody's age, on the other hand, would produce an ordinal set of data. Chi-square can be best applied to nominal data.

Summary

- Hypothesis testing is used to assess the plausibility of a hypothesis by using sample data.
- The null and alternative hypothesis are mutually exclusive, and only one can be true.
- In statistics, a Type I error is a false positive conclusion, while a Type II error is a false negative conclusion.
- The probability of making a Type I error is the significance level, or alpha (α), while the probability of making a Type II error is beta (β).
- P-values are used in hypothesis testing to help decide whether to reject the null hypothesis. The smaller the p-value, the more likely you are to reject the null hypothesis.
- ANOVA provides a statistical test of whether two or more population means are equal, and therefore generalizes the t-test beyond two means.
- A non-parametric alternative of ANOVA is PERMANOVA.
- There are three classes of models used in analysis of variance: Fixed effects models, random effects models and mixed effects models.
- One-way or two-way refers to the number of independent variables (IVs) in your Analysis of Variance test.
- MANOVA is just an ANOVA with several dependent variables.
- Chi-square tests are often used in hypothesis testing.

Keywords

- **Hypothesis testing:** It is an act in statistics whereby an analyst tests an assumption regarding a population parameter.

- **Null Hypothesis:** The null hypothesis is usually a hypothesis of equality between population parameters, e.g., a null hypothesis may state that the population mean return is equal to zero.
- **Alternate Hypothesis:** The alternative hypothesis is effectively the opposite of a null hypothesis (e.g., the population mean return is not equal to zero).
- **Type I error:** A Type I error means rejecting the null hypothesis when it's true.
- **Type II error:** A Type II error means not rejecting the null hypothesis when it's false.
- **P-value:** P-values are used in hypothesis testing to help decide whether to reject the null hypothesis.
- **ANOVA:** Analysis of variance (ANOVA) is a collection of statistical models and their associated estimation procedures (such as the "variation" among and between groups) used to analyze the differences among means.
- **Fixed-effects models:** The fixed-effects model (class I) of analysis of variance applies to situations in which the experimenter applies one or more treatments to the subjects of the experiment to see whether the response variable values change.
- **Random-effects model:** Random-effects model (class II) is used when the treatments are not fixed. This occurs when the various factor levels are sampled from a larger population.
- **Mixed-effects models:** A mixed-effects model (class III) contains experimental factors of both fixed and random-effect types, with appropriately different interpretations and analysis for the two types.
- **One-way ANOVA between groups:** It is used when you want to test two groups to see if there's a difference between them.
- **Two-way ANOVA without replication:** It is used when you have one group and you're double testing that same group.
- **Two-way ANOVA with replication:** Two groups, and the members of those groups are doing more than one thing.
- **Factorial ANOVA:** A factorial ANOVA is an Analysis of Variance test with more than one independent variable, or "factor".
- **Chi-square test:** A chi-square (χ^2) statistic is a test that measures how a model compares to actual observed data.

SelfAssessment

1. The smaller the p-value, the more likely you are to the null hypothesis.
A. Accept
B. Reject
C. Can't say
D. None of the above
2. If we are comparing just two things, then which test is more applicable?
A. T-test
B. ANOVA
C. P-Test
D. None of the above
3. Which test compares the size of any discrepancies between the expected results and the actual results?
A. Chi-square test
B. T-test
C. ANOVA
D. None of the above

4. Asking somebody their favorite color would produce a variable.
 - A. Nominal
 - B. Ordinal
 - C. Can't say
 - D. None of the above

5. Which of the following leads to false positive conclusion?
 - A. Type I error
 - B. Type II error
 - C. Type III error
 - D. None of the above

6. Which of the following leads to false negative conclusion?
 - A. Type I error
 - B. Type II error
 - C. Type III error
 - D. None of the above

7. The result says that "You have a fever, but actually you don't have". Then this is
 - A. Type I error
 - B. Type II error
 - C. Type III error
 - D. None of the above

8. The probability of making Type I error is
 - A. Alpha
 - B. Beta
 - C. Gamma
 - D. Lamda

9. The probability of making Type II error is
 - A. Alpha
 - B. Beta
 - C. Gamma
 - D. Lamda

10. The "one-way" or "two-way" ANOVA refers to
 - A. Number of dependent variables
 - B. Number of independent variables
 - C. Number of total variables
 - D. None of the above

11. What is nonparametric alternative of ANOVA?
 - A. PERMANOVA
 - B. NONPERMANOVA
 - C. ANOVANONPERM
 - D. None of the above

12. "You have a group of individuals randomly split into smaller groups and completing different tasks". You have this situation, in this case what will be applied?
 - A. One way ANOVA
 - B. Two-way ANOVA
 - C. Three-way ANOVA
 - D. None of the above

13. Which of the following usually represents the equality between the population parameters?
 - A. Null hypothesis
 - B. Alternate hypothesis

14. Can alternate hypothesis and null hypothesis be same at one time?
 - A. Yes
 - B. No

15. Which of the classes of models are used in analysis of variance?
- Fixed effects models
 - Random effects models
 - Mixed effects models
 - All of the above

Answer for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. B | 2. A | 3. A | 4. A | 5. A |
| 6. B | 7. A | 8. A | 9. B | 10. B |
| 11. A | 12. A | 13. A | 14. B | 15. D |

Review Questions

- What is hypothesis testing? What are two types of hypotheses?
- What are the steps of hypothesis testing? Explain with example and its mathematical representation.
- What are type I and type II errors? Explain its probabilities also. How can we find the trade off between Type I and Type II error?
- What is a P-value? How can we calculate the p-value? Write its importance.
- What is ANOVA? What are the classes of models used in ANOVA?
- What is ANOVA test? What does “one-way” or “two-way” ANOVA mean?
- Write the limitations of one-way ANOVA? Explain two-way ANOVA. Write the assumptions for two-way ANOVA.
- What is factorial ANOVA? Write the assumptions for factorial ANOVA. Also write its steps.
- What is chi-square test? Also write its formula. What does a Chi-Square Statistic Tell You?
- When to use a chi-square test? What is a chi-square test used for?



Further Readings

- <https://www.statisticshowto.com/probability-and-statistics/hypothesis-testing/>
- <https://byjus.com/maths/type-i-and-type-ii-errors/>



Web Links

- <https://www.statisticshowto.com/probability-and-statistics/hypothesis-testing/anova/>
- <https://www.statisticshowto.com/probability-and-statistics/chi-square/>

Unit 06: Machine Learning

CONTENTS

Objectives

Introduction

6.1 Components of Learning

6.2 How Machine Learning Works

6.3 Machine Learning Methods

6.4 Learning Problems

6.5 Designing a Learning System

6.6 Challenges in Machine Learning

Summary

Keywords

Self Assessment

Answer for Self Assessment

Review Questions

Further Readings

Objectives

After studying this unit, you will be able to

- understand the concept of machine learning,
- know the types of machine learning,
- understand the process of designing a learning system,
- understand the concept of learning task,
- understand the challenges in learning problems.

Introduction

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning is an important component of the growing field of data science. When machines assist humans, then machines should have a minimum level of intelligence. The challenge here is how machines could be incorporated with the level of intelligence required. Humans either learned through the hit and trial method or in the presence of some supervisors. Example: A child takes his finger to touch the fire of a candle that is lit. So, in a simple sense it is stated that the way a machine could learn about its tasks and objectives is called a machine learning.

Using statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics. As big data continues to expand and grow, the market demand for data scientists will increase, requiring them to assist in the identification of the most relevant business questions and subsequently the data to answer them.

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks T , as measured by P , improves with experience E . A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks T , as measured by P , improves with experience E .

- 1) Handwriting recognition learning problem
 - Task T: Recognizing and classifying handwritten words within images
 - Performance P: Percent of words correctly classified
 - Training experience E: A dataset of handwritten words with given classifications

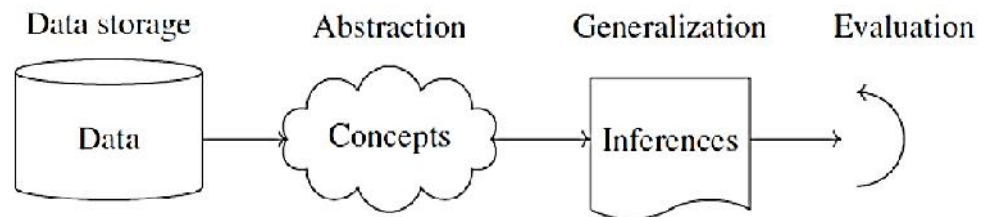
- 2) A robot driving learning problem
 - Task T: Driving on highways using vision sensor
 - Performance measure P: Average distance traveled before an error
 - Training experience: A sequence of images and steering commands recorded while observing a human driver

- 3) A chess learning problem
 - Task T: Playing chess
 - Performance measure P: Percent of games won against opponents
 - Training experience E: Playing practice games against itself

A computer program which learns from experience is called a machine learning program or simply a learning program. Such a program is sometimes also referred to as a learner.

6.1 Components of Learning

The learning process, whether by a human or a machine, can be divided into four components, namely, data storage, abstraction, generalization, and evaluation.



1. **Data storage:**Facilities for storing and retrieving huge amounts of data are an important component of the learning process. Humans and computers alike utilize data storage as a foundation for advanced reasoning.
 - In a human being, the data is stored in the brain and data is retrieved using electrochemical signals.
 - Computers use hard disk drives, flash memory, random access memory and similar devices to store data and use cables and other technology to retrieve data.
2. **Abstraction:**The second component of the learning process is known as abstraction. Abstraction is the process of extracting knowledge about stored data. This involves creating general concepts about the data. The creation of knowledge involves application of known models and creation of new models. The process of fitting a model to a dataset is known as training. When the model has been trained, the data is transformed into an abstract form that summarizes the original information.
3. **Generalization:**The third component of the learning process is known as generalization. The term generalization describes the process of turning the knowledge about stored data into a form that can be utilized for future action. These actions are to be carried out on tasks that are similar, but not identical, to those what have been seen before. In

generalization, the goal is to discover those properties of the data that will be most relevant to future tasks.

4. **Evaluation:** Evaluation is the last component of the learning process. It is the process of giving feedback to the user to measure the utility of the learned knowledge. This feedback is then utilized to effect improvements in the whole learning process.

6.2 How Machine Learning Works

The learning system of a machine learning algorithm is broken into three main parts.

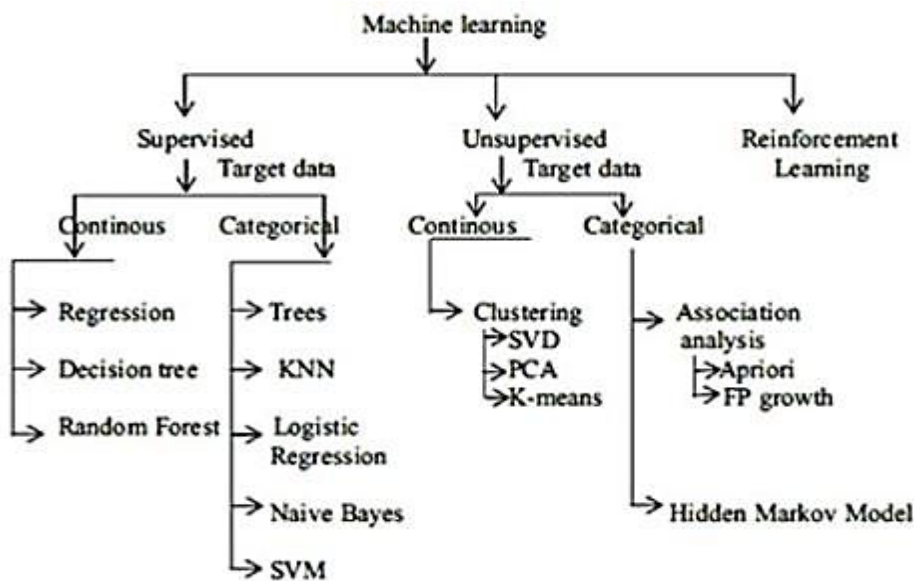
A Decision Process: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labelled or unlabeled, your algorithm will produce an estimate about a pattern in the data.

An Error Function: An error function serves to evaluate the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

An Model Optimization Process: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this evaluate and optimize process, updating weights autonomously until a threshold of accuracy has been met.


6.3 Machine Learning Methods

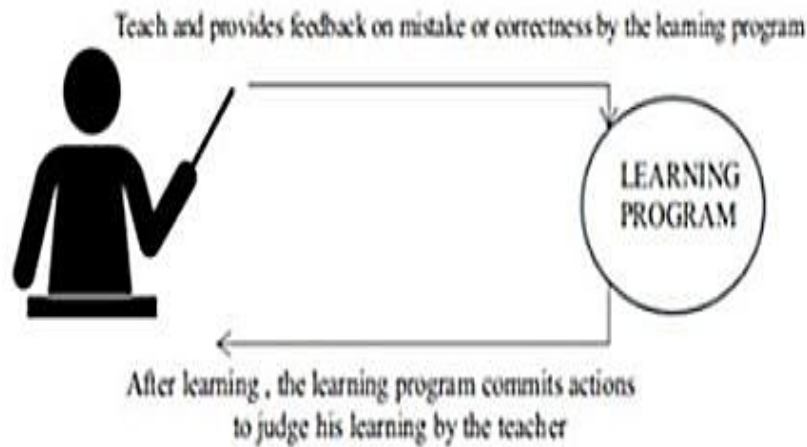
The machine learning methods are divided into three categories: supervised learning, unsupervised learning and reinforcement learning.



Supervised Learning


It is the field of learning where the machine learns with the help of a supervisor and instructor. A training set of examples with the correct responses (targets) is provided and, based on this training set, the algorithm generalises to respond correctly to all possible inputs. This is also called learning from exemplars. Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. In supervised learning, each example in the training set is a pair consisting of an input object (typically a vector) and an output value. A supervised learning algorithm analyses the training data and produces a function, which can be used for mapping new examples. In the optimal case, the function will correctly determine the class labels for unseen instances. Both classification and regression problems are supervised learning problems. A wide range of supervised learning algorithms are available, each with its strengths and weaknesses. There is no single learning algorithm that works best on all supervised learning problems.

 **Example:** Welcome Robot in home




Unsupervised Learning

Correct responses are not provided, but instead the algorithm tries to identify similarities between the inputs so that inputs that have something in common are categorised together. The statistical approach to unsupervised learning is known as density estimation. Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labelled responses. In unsupervised learning algorithms, a classification or categorization is not included in the observations. There are no output values and so there is no estimation of functions. Since the examples given to the learner are unlabelled, the accuracy of the structure that is output by the algorithm cannot be evaluated. The most common unsupervised learning method is cluster analysis, which is used for exploratory data analysis to find hidden patterns or grouping in data.

 **Example:** Clustering of different objects

Reinforcement Learning

This is somewhere between supervised and unsupervised learning. The algorithm gets told when the answer is wrong but does not get told how to correct it. It must explore and try out different possibilities until it works out how to get the answer right. Reinforcement learning is sometime called learning with a critic because of this monitor that scores the answer but does not suggest improvements. Reinforcement learning is the problem of getting an agent to act in the world to maximize its rewards. A learner (the program) is not told what actions to take as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situations and, through that, all subsequent rewards. Every state and action have a reward: positive and negative. The reinforcement process is accomplished by an agent called a reinforcement agent.

 **Example:** Learning of four-wheel vehicle

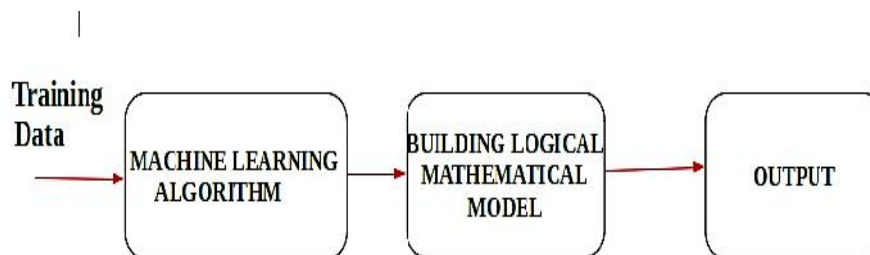
6.4 Learning Problems

- 1) Identification of spams
- 2) Recommending Products
- 3) Customer segmentation

- 4) Image and video recognition
- 5) Fraudulent Transactions
- 6) Demand Forecasting
- 7) Virtual Personal Assistant
- 8) Sentiment Analysis
- 9) Customer Service Automation

6.5 Designing a Learning System

According to Arthur Samuel "Machine Learning enables a Machine to Automatically learn from Data, improve performance from an Experience and predict things without explicitly programmed." When we fed the Training Data to Machine Learning Algorithm, this algorithm will produce a mathematical model and with the help of the mathematical model, the machine will make a prediction and take a decision without being explicitly programmed. Also, during training data, the more machine will work with it the more it will get experience and the more it will get experience the more efficient result is produced.



Learn from
Data



Choose training experience

The very important and first task is to choose the training data or training experience which will be fed to the Machine Learning Algorithm. It is important to note that the data or experience that we fed to the algorithm must have a significant impact on the Success or Failure of the Model. So, training data or experience should be chosen wisely.

Choose target function

It means according to the knowledge fed to the algorithm the machine learning will choose NextMove function which will describe what type of legal moves should be taken.

Choosing representation of a target function

Data Science Toolbox

When the machine algorithm will know all the possible legal moves the next step is to choose the optimized move using any representation, i.e., using linear Equations, Hierarchical Graph Representation, Tabular form etc. The NextMove function will move the Target move like out of these moves which will provide more success rate.

Choosing function approximation algorithm

An optimized move cannot be chosen just with the training data. The training data had to go through with set of examples and through these examples the training data will approximate which steps are chosen and after that machine will provide feedback on it.

Final design

The final design is created at last when system goes from number of examples , failures and success, correct and incorrect decision and what will be the next step etc.

6.6 Challenges in Machine Learning

- 1) Poor quality of data
- 2) Underfitting of training data
- 3) Overfitting of training data
- 4) Machine learning is a complex process
- 5) Lack of training data
- 6) Slow implementation

Summary

- Machine learning is programming computers to optimize a performance criterion using example data or experience.
- A computer program which learns from experience is called a machine learning program.
- The learning process, whether by a human or a machine, can be divided into four components, namely, data storage, abstraction, generalization, and evaluation.
- For any learning system, we must be knowing the three elements — **T (Task)**, **P (Performance Measure)**, and **E (Training Experience)**.
- Reinforcement learning is somewhere between supervised and unsupervised learning.

Keywords

- **Abstraction:** Abstraction is the process of extracting knowledge about stored data. This involves creating general concepts about the data as a whole.
- **Generalization:** The term generalization describes the process of turning the knowledge about stored data into a form that can be utilized for future action.
- **Evaluation:** It is the process of giving feedback to the user to measure the utility of the learned knowledge. This feedback is then utilized to effect improvements in the whole learning process.
- **Supervised Learning:** Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.
- **Unsupervised Learning:** Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.

SelfAssessment

1. Which of the following supervised machine learning algorithms is not applicable on continuous data?
 - A. Regression
 - B. Decision tree
 - C. Random Forest
 - D. KNN

2. Find the odd one out.
 - A. K-means
 - B. SVM
 - C. Naïve Bayes
 - D. Random Forest

3. For association analysis, the techniques can be
 - A. Apriori algorithm
 - B. FP growth
 - C. Both of the above
 - D. None of the above

4. Which of the following supervised machine learning algorithms is not applicable on categorical data?
 - A. Trees
 - B. KNN
 - C. SVM
 - D. Random forest

5. Find the odd one out.
 - A. SVD
 - B. PCA
 - C. K-means
 - D. SVM

6. In which type of machine learning, the column of label is not provided along with data?
 - A. Supervised
 - B. Unsupervised
 - C. Can't say
 - D. None of the above

7. Reinforcement learning works best with ... environment.
 - A. Static
 - B. Dynamic
 - C. Can't say
 - D. None of the above

8. In unsupervised machine learning, for target data which techniques can be applied?
 - A. Clustering
 - B. Association rule mining
 - C. Hidden Markov model
 - D. All of the above

9. In which case, the feedback is provided on mistake, and it is corrected by learning process?
 - A. Supervised
 - B. Unsupervised
 - C. Can't say
 - D. None of the above

10. Which of the following usually interacts with the environment?
 - A. Supervised machine learning
 - B. Unsupervised machine learning
 - C. Reinforcement learning

Data Science Toolbox

- D. None of the above
11. What are the challenges for machine learning algorithms?
 A. Lack of training data
 B. Slow implementation
 C. Poor quality of data
 D. All of the above
12. The field of learning where the machine learns with the help of an instructor is known as
 A. Supervised machine learning
 B. Unsupervised machine learning
 C. Non supervised machine learning
 D. None of the above
13. In this kind of learning, the learning happens with an interaction with the environment. This is known as
 A. Supervised machine learning
 B. Unsupervised machine learning
 C. Reinforcement learning
 D. None of the above
14. On which learning problem, the machine learning algorithms can be applied?
 A. Sentiment analysis
 B. Image and video recognition
 C. Identification of spams
 D. All of the above
15. The machine learning type can be
 A. Supervised machine learning
 B. Unsupervised machine learning
 C. Reinforcement learning
 D. All of the above

Answer for Self Assessment

1. D 2. A 3. C 4. D 5. D
 6. B 7. B 8. D 9. A 10. C
 11. D 12. A 13. C 14. D 15. D

Review Questions

1. What is machine learning? Explain the concept of learning with an example.
2. What are the types of machine learning? Explain with example.
3. Explain the components of learning process in detail. Also explain how machine learning works.
4. Give few examples of learning problems. Also explain how to design a learning system.
5. What are the challenges in machine learning? Also explain how we can overcome these challenges.

**Further Readings**

<https://www.ibm.com/in-en/cloud/learn/machine-learning>



Web Links

<https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML>

Unit 07: Unsupervised Learning

CONTENTS

Objectives

Introduction

7.1 Unsupervised Learning

7.2 Clustering

7.3 Partitioning Clustering

7.4 Performance Measures

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this lecture, students will:

- Understand the unsupervised learning
- Understanding the clustering algorithms
- Understanding k-means algorithm
- Understand k-mode algorithm
- Understand k-median algorithm
- Know the performance metrics of clustering

Introduction

Unsupervised learning is a machine learning technique in which models are not supervised using training dataset. Instead, models itself find the hidden patterns and insights from the given data.

7.1 Unsupervised Learning

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.

Why use unsupervised learning?

There are various benefits to use unsupervised learning. These are:

- Unsupervised learning is helpful for finding useful insights from the data.
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
- Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.

- In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.

Advantages of Unsupervised Learning

The unsupervised learning is advantageous in many ways. These are:

- Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.
- Unsupervised learning is preferable as it is easy to get unlabeled data in comparison to labeled data.

Disadvantages of Unsupervised learning

Along with many advantages, it provides few disadvantages as well:

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.

Types of unsupervised learning

There are two main types of unsupervised learning. These are: clustering and association.

Clustering

Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.

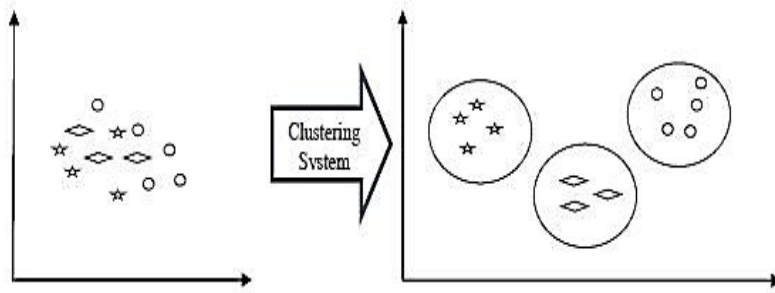
Association

An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is Market Basket Analysis.

7.2 Clustering

Clustering methods are one of the most useful unsupervised ML methods. Clustering is the task of grouping a set of customers in such a way that customers in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). These methods are used to find similarity as well as the relationship of patterns among data samples and then cluster those samples into groups having similarity based on features. Cluster analysis uses mathematical models to discover groups of similar customers based on the smallest variations among customers within each group. A Clustering Algorithm tries to analyze natural groups of data based on some similarity. Clustering is important because it determines the intrinsic grouping among the present unlabeled data. They basically make some assumptions about data points to constitute their similarity. Clustering is dividing data points into homogeneous classes or clusters:

- Points in the same group are as similar as possible
- Points in different group are as dissimilar as possible



Applications of Clustering

- 1) **Data summarization and compression** – Clustering is widely used in the areas where we require data summarization, compression, and reduction as well. The examples are image processing and vector quantization.
- 2) **Collaborative systems and customer segmentation** – Since clustering can be used to find similar products or same kind of users, it can be used in collaborative systems and customer segmentation.
- 3) **Serve as a key intermediate step for other data mining tasks** – Cluster analysis can generate a compact summary of data for classification, testing, hypothesis generation; hence, it serves as a key intermediate step for other data mining tasks also.
- 4) **Trend detection in dynamic data** – Clustering can also be used for trend detection in dynamic data by making various clusters of similar trends.
- 5) **Social network analysis** – Clustering can be used in social network analysis. The examples are generating sequences in images, videos, or audios.
- 6) **Biological data analysis** – Clustering can also be used to make clusters of images, videos hence it can successfully be used in biological data analysis.

7.3 Partitioning Clustering

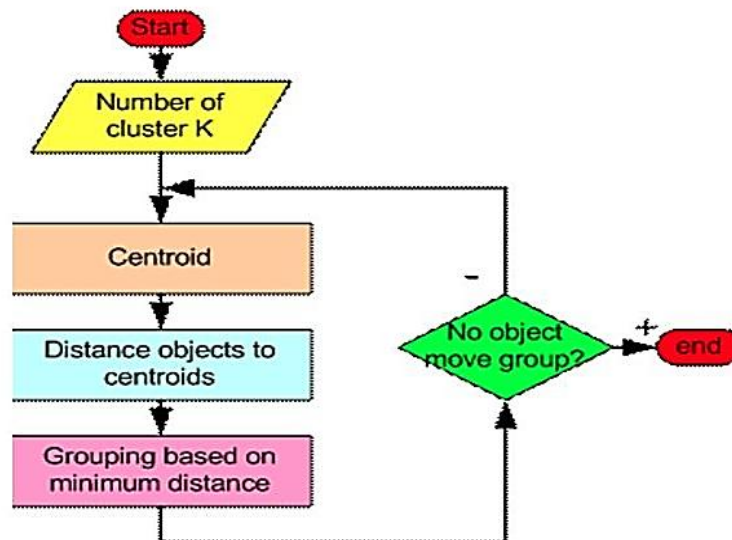
The most popular class of clustering algorithms that we have is the iterative relocation algorithms. These algorithms minimize a given clustering criterion by iteratively relocating data points between clusters until a (locally) optimal partition is attained.

K-Means algorithm

It is one of the most used algorithms for partitioning a given data set into a set of k groups (i.e., k clusters), where k represents the number of groups. It classifies objects in multiple groups (i.e., clusters), such that objects within the same cluster are as similar as possible (i.e., high intra-class similarity), whereas objects from different clusters are as dissimilar as possible (i.e., low inter-class similarity). In k -means clustering, each cluster is represented by its center (i.e., centroid) which corresponds to the mean of points assigned to the cluster. The basic idea behind k -means clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is minimized.

Basic Phenomenon

Given a set of observations (x_1, x_2, \dots, x_n) , where each observation is a d -dimensional real vector, k -means clustering aims to partition the n observations into k ($k \leq n$) sets $S = \{S_1, S_2, \dots, S_k\}$ to minimize the within-cluster sum of squares (WCSS) (i.e., variance).



- Step - 1

The first step when using k-means clustering is to indicate the number of clusters (k) that will be generated in the final solution.

- Step - 2

The algorithm starts by randomly selecting k objects from the data set to serve as the initial centers for the clusters. The selected objects are also known as cluster means or centroids.

- Step - 3

Next, each of the remaining objects is assigned to its closest centroid, where closest is defined using the Euclidean distance between the object and the cluster mean. This step is called “cluster assignment step”.

- Step - 4

After the assignment step, the algorithm computes the new mean value of each cluster. The term cluster “centroid update” is used to design this step. Now that the centers have been recalculated, every observation is checked again to see if it might be closer to a different cluster. All the objects are reassigned again using the updated cluster means.

- Step - 5

The cluster assignment and centroid update steps are iteratively repeated until the cluster assignments stop changing (i.e., until convergence is achieved). That is, the clusters formed in the current iteration are the same as those obtained in the previous iteration.

- Given an initial set of k means $m_1^{(1)}, \dots, m_k^{(1)}$, the algorithm proceeds by alternating between two steps:

1) **Assignment step:** Assign each observation to the cluster with the nearest mean: that with the least squared Euclidean distance.

2) **Update step:** Recalculate means (centroids) for observations assigned to each cluster.

Specific features of k-means algorithm

- It is an efficient algorithm for clustering data items, i.e., the Euclidean distance is used as a metric and variance is used as a measure of cluster scatter.
- It is very sensitive to outliers.
- In k-means algorithm, the number of clusters need to be specified in advance. The number of clusters k is an input parameter: an inappropriate choice of k may yield poor results. That is

why, when performing k-means, it is important to run diagnostic checks for determining the number of clusters in the data set.

- The convergence to a local minimum may produce wrong results.

Applications of k-means algorithm

- K-means clustering is rather easy to apply to even large data sets.
- It has been successfully used in market segmentation, computer vision, and astronomy among many other domains.
- It often is used as a preprocessing step for other algorithms, for example to find a starting configuration.
- K-means originates from signal processing, and still finds use in this domain. For example, in computer graphics, color quantization is the task of reducing the color palette of an image to a fixed number of colors k . The K-means algorithm can easily be used for this task and produces competitive results
- In cluster analysis, the k-means algorithm can be used to partition the input data set into k partitions (clusters).

K-Mode algorithm

K-Modes clustering is one of the unsupervised Machine Learning algorithms that is used to cluster categorical variables.

Why not k-means algorithm?

- K-Means uses mathematical measures (distance) to cluster continuous data.
- The lesser the distance, the more similar our data points are.
- Centroids are updated by Means.
- But for categorical data points, we cannot calculate the distance. So, we go for K Modes algorithm. It uses the dissimilarities (total mismatches) between the data points.
- The lesser the dissimilarities the more similar our data points are. It uses Modes instead of means.

Algorithm

1. Pick K observations at random and use them as leaders/clusters
2. Calculate the dissimilarities and assign each observation to its closest cluster
3. Define new modes for the clusters
4. Repeat 2-3 steps until there is no re-assignment required



Example

Imagine we have a dataset that has the information about hair color, eye color, and skin color of persons. We aim to group them based on the available information. Hair color, eye color, and skin color are all categorical variables.

Person	Hair Color	Eye Color	Skin Color
P1	Blonde	Amber	Fair

Data Science Toolbox

P2	Brunette	Gray	Brown
P3	Red	Green	Brown
P4	Black	Hazel	Brown
P5	Brunette	Amber	Fair
P6	Black	Gray	Brown
P7	Red	Green	Fair
P8	Black	Hazel	Fair

Step 1: Pick k observations at random

Leaders			
P1	Blonde	Amber	Fair
P7	Red	Green	Fair
P8	Black	Hazel	Fair

Step 2: Calculate the dissimilarities

Iteratively compare the cluster data points to each of the observations. In the first iteration, compare P1 of leaders with P1 of data given. Similar data points give 0, dissimilar data points give 1.

Leaders			
P1	Blonde	Amber	Fair
P7	Red	Green	Fair
P8	Black	Hazel	Fair
Person	Hair Color	Eye Color	Skin Color
P1	Blonde	Amber	Fair
P2	Brunette	Gray	Brown
P3	Red	Green	Brown
P4	Black	Hazel	Brown
P5	Brunette	Amber	Fair
P6	Black	Gray	Brown
P7	Red	Green	Fair
P8	Black	Hazel	Fair

Comparing leader/Cluster P1 to the observation P1 gives 0 dissimilarities. In next iteration, compare P1 of leaders table with the table data given. Comparing leader/cluster P1 to the

Unit 07: Unsupervised Learning

observation P2 gives 3(1+1+1) dissimilarities. Likewise, calculate all the dissimilarities and put them in a matrix and assign the observations to their closest cluster (cluster that has the least dissimilarity). So, with this we will calculate the matrix for cluster 1, 2 and 3 which is shown below.

Person	Cluster 1 (P1)
P1	0
P2	3
P3	3
P4	3
P5	1
P6	3
P7	2
P8	2

Person	Cluster 2 (P7)
P1	2
P2	3
P3	1
P4	3
P5	2
P6	3
P7	0
P8	2

Person	Cluster 3 (P8)
P1	2

P2	3
P3	0
P4	1
P5	2
P6	2
P7	2
P8	0

- After step 2, the observations P1, P2, P5 are assigned to cluster 1; P3, P7 are assigned to Cluster 2; and P4, P6, P8 are assigned to cluster 3.
- If all the clusters have the same dissimilarity with an observation, assign to any cluster randomly. In our case, the observation P2 has 3 dissimilarities with all the leaders. I randomly assigned it to Cluster 1.

Dissimilarity Matrix

	Cluster 1 (P1)	Cluster 2 (P7)	Cluster 3 (P8)	Cluster
P1	0	2	2	Cluster 1
P2	3	3	3	Cluster 1
P3	3	1	3	Cluster 2
P4	3	3	1	Cluster 3
P5	1	2	2	Cluster 1
P6	3	3	2	Cluster 3
P7	2	0	2	Cluster 2
P8	2	2	0	Cluster 3

Step 3: Define new modes for the clusters

- Mode is simply the **most observed value**.
- Mark the observations according to the cluster they belong to. Observations of Cluster 1 are marked in Yellow, Cluster 2 are marked in Brick red, and Cluster 3 are marked in Purple.
- Cluster 1 observations(P1, P2, P5) has brunette as the most observed hair color, amber as the most observed eye color, and fair as the most observed skin color.

Unit 07: Unsupervised Learning

person	hair color	eye color	skin color
P1	blonde	amber	fair
P2	brunette	gray	brown
P3	red	green	brown
P4	black	hazel	brown
P5	brunette	amber	fair
P6	black	gray	brown
P7	red	green	fair
P8	black	hazel	fair

New Leaders			
	Hair Color	Eye Color	Skin Color
Cluster 1	Brunette	Amber	Fair
Cluster 2	Red	Green	Fair
Cluster 3	Black	Hazel	Brown

Repeat steps 2-4

- After obtaining the new leaders, again calculate the dissimilarities between the observations and the newly obtained leaders.

New Leaders			
	hair color	eye color	skin color
Cluster 1	brunette	amber	fair
Cluster 2	red	green	fair
Cluster 3	black	hazel	brown

person	hair color	eye color	skin color
P1	blonde	amber	fair
P2	brunette	gray	brown
P3	red	green	brown
P4	black	hazel	brown
P5	brunette	amber	fair
P6	black	gray	brown
P7	red	green	fair
P8	black	hazel	fair

- Comparing Cluster 1 to the observation P1 gives 1 dissimilarity.

New Leaders			
	hair color	eye color	skin color
Cluster 1	brunette	amber	fair
Cluster 2	red	green	fair
Cluster 3	black	hazel	brown

person	hair color	eye color	skin color
P1	blonde	amber	fair
P2	brunette	gray	brown
P3	red	green	brown
P4	black	hazel	brown
P5	brunette	amber	fair
P6	black	gray	brown
P7	red	green	fair
P8	black	hazel	fair

- Comparing Cluster 1 to the observation P2 gives 2 dissimilarities.
- Likewise, calculate all the dissimilarities and put them in a matrix. Assign each observation to its closest cluster.

	Cluster 1	Cluster 2	Cluster 3	Cluster
P1	1 ✓	2	3	Cluster 1
P2	2 ✓	3	2	Cluster 1
P3	3	1 ✓	2	Cluster 2
P4	3	3	0 ✓	Cluster 3
P5	0 ✓	2	3	Cluster 1
P6	3	3	1 ✓	Cluster 3
P7	2	0 ✓	3	Cluster 2
P8	2	2	1 ✓	Cluster 3

K-Median Algorithm

It is a variation of k-means clustering where instead of calculating the mean for each cluster to determine its centroid, one instead calculates the median. In this, we use L1 norm as the distance measure. The k-medians approach to clustering data attempts to minimize the 1-norm distances between each point and its closest cluster center. This minimization of distances is obtained by setting the center of each cluster to be the median of all points in that cluster.

Benefits over k-means

- The mean is a measurement that is highly vulnerable to outliers. Even just one drastic outlier can pull the value of the mean away from the majority of the data set, which can be a high concern when operating on very large data sets.
- The median, on the other hand, is a statistic incredibly resistant to outliers, for in order to deter the median away from the bulk of the information, it requires at least 50% of the data to be contaminated

Criterion function

- The criterion function for k-median algorithm is:

$$S = \sum_{k=1}^k \sum_x |x_{ij} - med_{kj}|$$

Algorithm

1. Select K points as the initial representative objects (i.e., as initial k medians).
2. Repeat
 1. Assign each point to its nearest median
 2. Re-compute the median using the median of each individual feature
3. Until convergence criterion is satisfied

7.4 Performance Measures

The most popular metrics evaluation metrics for clustering algorithms are the Silhouette coefficient, Dunn's Index and Rand Index.

Silhouette Coefficient

$$S = \frac{b - a}{\max(a, b)}$$

The Silhouette Coefficient is defined for each sample and is composed of two scores:

a: The mean distance between a sample and all other points in the same cluster.

b: The mean distance between a sample and all other points in the next nearest cluster.

Score -1: Incorrect clustering

Score +1: Highly dense clustering

Score around 0: Overlapping clustering

The score is higher when clusters are dense and well separated, which relates to a standard concept of a cluster.

Dunn's Index

Dunn's Index is equal to the minimum inter-cluster distance divided by the maximum cluster size. The large inter-cluster distances (better separation) and smaller cluster sizes (more compact clusters) lead to a higher DI value. A higher DI implies better clustering. It assumes that better clustering means that clusters are compact and well-separated from other clusters.

Rand Index

It computes a similarity measure between two clusters by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clustering.

Summary

- Unsupervised learning is a machine learning technique in which models are not supervised using training dataset.
- Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data.
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.
- Clustering methods are one of the most useful unsupervised ML methods.
- Clustering is dividing data points into homogeneous classes or clusters with the condition: The points in the same group are as similar as possible and points in different group are as dissimilar as possible.

- The basic idea behind k-means clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is minimized.
- We use L1 norm as the distance measure for k-median algorithm.
- The most popular metrics evaluation metrics for clustering algorithms are the Silhouette coefficient, Dunn's Index and Rand Index.

Keywords

- **Unsupervised learning:** The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.
- **Clustering:** Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group.
- **Cluster analysis:** Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.
- **Association Rule:** An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database.
- **K-means algorithm:** It classifies objects in multiple groups (i.e., clusters), such that objects within the same cluster are as similar as possible (i.e., high intra-class similarity), whereas objects from different clusters are as dissimilar as possible (i.e., low inter-class similarity).
- **K-mode algorithm:** It is one of the unsupervised Machine Learning algorithms that is used to cluster **categorical variables**.
- **K-median algorithm:** It is a variation of k-means clustering where instead of calculating the mean for each cluster to determine its centroid, one instead calculates the median.

SelfAssessment

1. A machine learning technique in which models are not supervised using training dataset is known as:
 - A. Unsupervised learning
 - B. Supervised learning
 - C. Non defined learning
 - D. None of the above

2. Unsupervised learning cannot be directly applied to
 - A. Classification problems
 - B. Regression problems
 - C. Both of the above
 - D. None of the above

3. The important types of unsupervised algorithms are:
 - A. Association rule
 - B. Clustering
 - C. Both of the above
 - D. None of the above

-
4. is an unsupervised learning method which is used for finding the relationships between variables in the large database.
- A. Association rule
 - B. Clustering
 - C. Both of the above
 - D. None of the above
5. What is the task of grouping a set of customers in such a way that customers in the same group are more like each other than to those in other groups?
- A. Association rule
 - B. Clustering
 - C. Both of the above
 - D. None of the above
6. What are the applications of clustering?
- A. Data summarization and compression
 - B. Trend detection in dynamic data.
 - C. Biological data analysis
 - D. All of the above
7. The basic idea behind k-means clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is
- A. Minimized
 - B. Maximized
 - C. Remains same
 - D. None of the above
8. The basic idea behind clustering consists of defining clusters so that the total intra-cluster variation (known as total within-cluster variation) is minimized.
- A. K-means algorithm
 - B. K-mode algorithm
 - C. K-median algorithm
 - D. None of the above
9. The first step in k-means algorithm is to define the value of k. This statement is
- A. True
 - B. False
10. clustering is one of the unsupervised Machine Learning algorithms that is used to cluster categorical variables.
- A. K-means algorithm
 - B. K-mode algorithm

- C. K-median algorithm
 - D. None of the above
11. In k-modes algorithm, the Between the data points are used.
- A. Similarities
 - B. Dissimilarities
 - C. Half of the original points
 - D. None of the above
12. In k-median algorithm, what is used as a distance measure?
- A. L1 norm
 - B. L2 norm
 - C. L0 norm
 - D. None of the above
13. Which algorithm is more resistant to outliers in data?
- A. K-means algorithm
 - B. K-mode algorithm
 - C. K-median algorithm
 - D. None of the above
14. Which of the following are the performance measures for clustering algorithms?
- A. Silhouette coefficient
 - B. Dunn's Index
 - C. Rand Index
 - D. All of the above
15. In Silhouette coefficient, if the score is +1. Then it represents
- A. Incorrect clustering
 - B. Highly dense clustering
 - C. Overlapping clustering
 - D. None of the above

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. A | 2. C | 3. C | 4. A | 5. B |
| 6. D | 7. A | 8. A | 9. A | 10. B |
| 11. B | 12. A | 13. C | 14. D | 15. B |

Review Questions

1. What is unsupervised learning? Write down its advantages and disadvantages.
2. What are the applications of unsupervised learning? Also explain what the benefits are of using unsupervised learning.
3. What are the types of clustering? Explain all in detail.
4. What is k-means algorithm? Explain its basic phenomenon and specific features.
5. What is k-mode algorithm? Why is it preferred more over k-means algorithm? Explain with one example.
6. What is k-median algorithm? Explain its criterion function and algorithm.
7. What are the performance measures of clustering algorithms?



Further Readings

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>



Web Links

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Unit 08: Supervised Learning

CONTENTS

Objectives

Introduction

8.1 Supervised Learning

8.2 Classification

8.3 K-NN Algorithm

8.4 Naïve Bayes

8.5 Cross-Validation

8.6 Metrics of Classification Algorithms

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this unit, you will be able to understand:

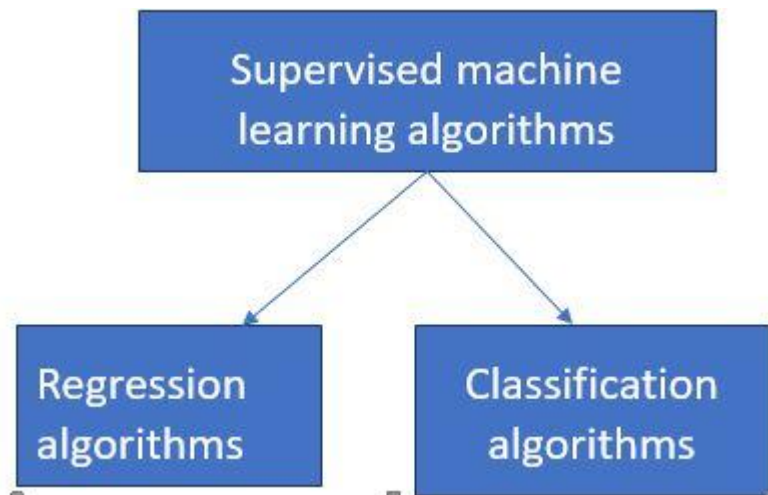
- Understand the meaning of classification
- Understand the KNN algorithm
- Understand the Naïve Bayes algorithm
- Understand the cross-validation
- Understand the performance metrics of classification algorithms

Introduction

Supervised learning is the types of machine learning in which machines are trained using well "labeled" training data, and on basis of that data, machines predict the output. The labeled data means some input data is already tagged with the correct output.

8.1 Supervised Learning

In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y). In the real-world, supervised learning can be used for Risk Assessment, Image classification, Fraud Detection, spam filtering, etc.



8.2 Classification

Classification: It is the operation of separating various entities into several classes.

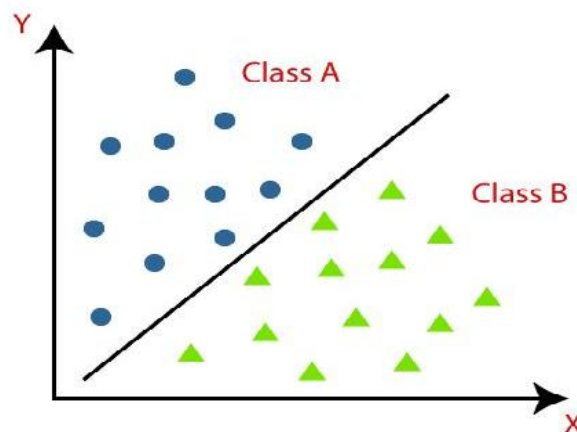
Classification algorithm: The Classification algorithm is a supervised learning technique that is used to identify the category of new observations based on training data.

In Classification, a program learns from the given dataset or observations and then classifies new observation into several classes or groups such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories. The output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.

Output function: In classification algorithm, a discrete output function(y) is mapped to input variable(x). $y=f(x)$, where y = categorical output.



Example



Types of classification

- **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier. Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier. Example: Classifications of types of crops, Classification of types of music.

Learning in classification problems

- **Lazy Learners:** Lazy Learner firstly stores the training dataset and wait until it receives the test dataset. In Lazy learner case, classification is done on the basis of the most related data stored in the training dataset. It takes less time in training but more time for predictions. Example: K-NN algorithm, Case-based reasoning.
- **Eager Learners:** Eager Learners develop a classification model based on a training dataset before receiving a test dataset. Opposite to Lazy learners, Eager Learner takes more time in learning, and less time in prediction. Example: Decision Trees, Naïve Bayes, ANN.

Types of ML classification algorithms

- Linear Models
 - Logistic Regression
 - Support Vector Machines
- Non-linear Models
 - K-Nearest Neighbors
 - Kernel SVM
 - Naïve Bayes
 - Decision Tree Classification
 - Random Forest Classification

Basic terminologies

- **Classifier:** An algorithm that maps the input data to a specific category.
- **Classification model:** It tries to draw some conclusion from the input values given for training. It will predict the class labels/categories for the new data.
- **Feature:** It is an individual measurable property of a phenomenon being observed.
- **Binary Classification:** Yes/No
- **Multi-class classification:** An animal can be a cat or dog but not both at the same time.
- **Multi-label classification:** A news article can be about sports, a person, and location at the same time.

Steps in building a classification model

1. Initialize
2. Train the classifier
3. Predict the target
4. Evaluate

Applications of classification algorithms

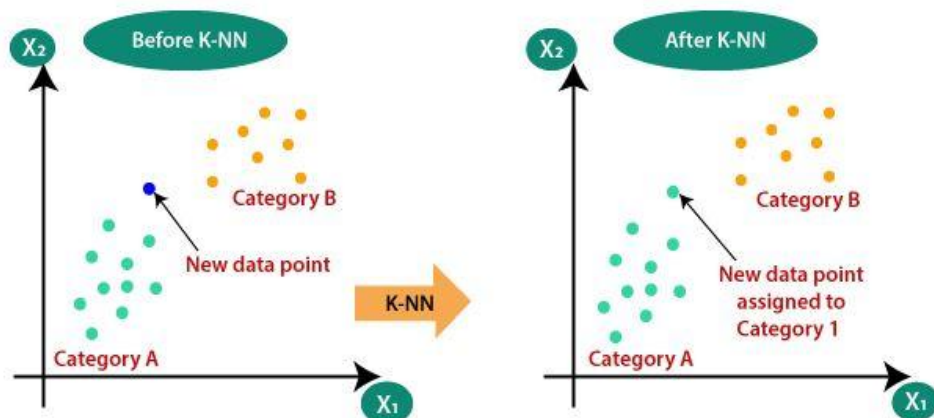
1. Sentiment analysis - It is a machine learning text analysis technique that assigns sentiment (opinion, feeling, or emotion) to words within a text, or an entire text, on a polarity scale of Positive, Negative, or Neutral.
2. Email spam classification - Email applications use the above algorithms to calculate the likelihood that an email is either not intended for the recipient or unwanted spam.
3. Document classification - Document classification is the ordering of documents into categories according to their content.
4. Image classification - Image classification assigns previously trained categories to a given image.
5. Diagnosis of disease - Based upon the gene/values, detects a person is suffering from the disease or not.

8.3 K-NN Algorithm

It is a supervised learning technique. It works on similarity measure. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most like the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm. K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data. It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much like the new data.



Need



Working

Step-1: Select the number K of the neighbors

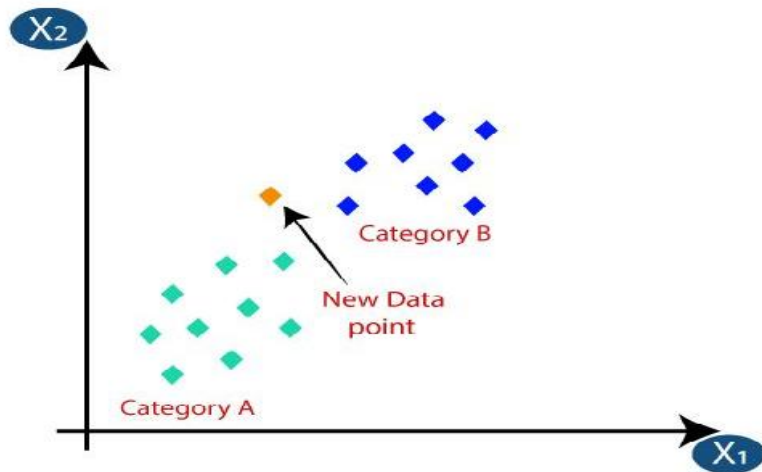
Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

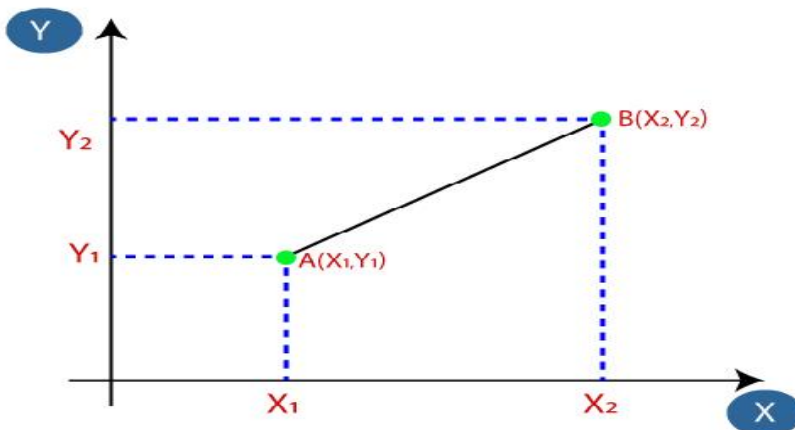
Step-6: The model is ready.



- Firstly, we will choose the number of neighbors,

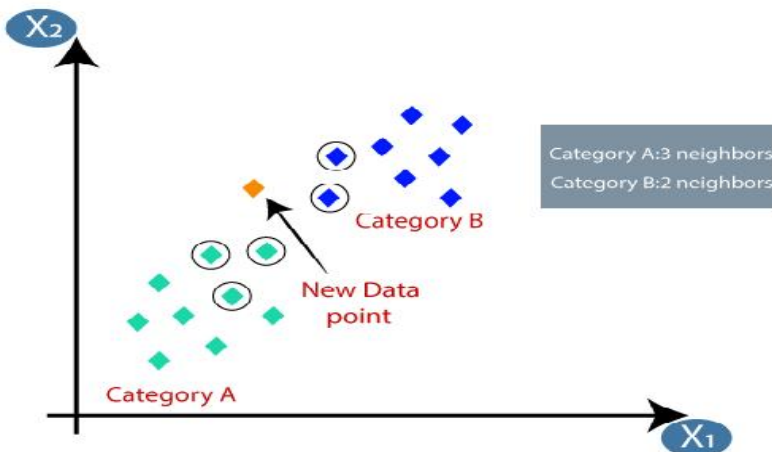
Let us suppose $k=5$.

- Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points. It can be calculated as:



$$\text{Euclidean Distance between } A_1 \text{ and } B_2 = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$$

- By calculating the Euclidean distance, we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.



- 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Selection of K

- There is no way to determine the best value for "K", so we need to try some values to find the best out of them.
- A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.

Advantages of KNN algorithms

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN algorithm

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

8.4 Naïve Bayes

It is a supervised learning algorithm. The naïve Bayes algorithm for classification is based on Bayes theorem. It is mainly used in text classification that includes a high-dimensional training dataset. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts based on the probability of an object. The popular examples are spam filtration, sentiment analysis and classifying articles. It is composed of two words - Naïve and Bayes. Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.

Bayes Theorem

This is called Bayes because it depends on the principle of Bayes' Theorem. Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**. It is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

P(A | B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B | A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Steps of Naïve Bayes

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
3. Now, use Bayes theorem to calculate the posterior probability.



Example

Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**".

Dataset given is

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Frequency table

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

Likelihood table

Weather	No	Yes	
Overcast	0	5	$5/14 = 0.35$
Rainy	2	2	$4/14 = 0.29$
Sunny	2	3	$5/14 = 0.35$
All	$4/14 = 0.29$	$10/14 = 0.71$	

Applying Bayes theorem

$$P(\text{Yes} | \text{Sunny}) = P(\text{Sunny} | \text{Yes}) * P(\text{Yes}) / P(\text{Sunny})$$

$$P(\text{Sunny} | \text{Yes}) = 3/10 = 0.3$$

$$P(\text{Sunny}) = 0.35$$

$$P(\text{Yes}) = 0.71$$

$$\text{So, } P(\text{Yes} | \text{Sunny}) = 0.3 * 0.71 / 0.35 = \mathbf{0.60}$$

$$P(\text{No} | \text{Sunny}) = P(\text{Sunny} | \text{No}) * P(\text{No}) / P(\text{Sunny})$$

$$P(\text{Sunny} | \text{NO}) = 2/4 = 0.5$$

$$P(\text{No}) = 0.29$$

$$P(\text{Sunny}) = 0.35$$

$$\text{So, } P(\text{No} | \text{Sunny}) = 0.5 * 0.29 / 0.35 = \mathbf{0.41}$$

So, as we can see from the above calculation that $P(\text{Yes} | \text{Sunny}) > P(\text{No} | \text{Sunny})$.

Hence on a Sunny day, Player can play the game.

Advantages

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

Disadvantages

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications

- It is used for **Credit Scoring**.
- It is used in **medical data classification**.
- It can be used in **real-time predictions** because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as **Spam filtering** and **Sentiment analysis**.

8.5 Cross-Validation

Cross validation is a model evaluation method. It is one of the most widely used data resampling methods to assess the generalization ability of a predictive model. The objective is to not use the entire data set when training a learner. Some of the data is removed before training begins. Then when training is done, the data that was removed can be used to test the performance of the learned model on "new" data. There are three main types of cross validation. These are holdout validation, k-fold cross validation and LOOCV.

Holdout validation: The holdout technique is an exhaustive cross-validation method that randomly splits the dataset into train and test data depending on data analysis. (Image by Author), 70:30 split of Data into training and validation data respectively.

K-Fold cross validation: K-fold Cross-Validation is when the dataset is split into a K number of folds and is used to evaluate the model's ability when given new data. K refers to the number of groups the data sample is split into. For example, if you see that the k-value is 5, we can call this a 5-fold cross-validation.

LOOCV: LOOCV (Leave One Out Cross-Validation) is a type of cross-validation approach in which each observation is considered as the validation set and the rest (N-1) observations are considered as the training set. In LOOCV, fitting of the model is done and predicting using one observation validation set.

8.6 Metrics of Classification Algorithms

There are various measures of classification algorithms. The important measures are accuracy, precision, recall, F1 score AUC-ROC. These all measures can be easily calculated using confusion matrix.

Confusion matrix

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

From confusion matrix we can calculate accuracy, precision, sensitivity, specificity, recall and AUC-ROC.

Accuracy: Accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition: Accuracy = Number of correct predictions/ Total number of predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision:

Precision is one indicator of a machine learning model's performance - the quality of a positive prediction made by the model. Precision refers to the number of true positives divided by the total number of positive predictions (i.e., the number of true positives plus the number of false positives).

$$Precision = \frac{True\ positive}{True\ positive + False\ positive}$$

Recall

The recall is calculated as **the ratio between the numbers of Positive samples correctly classified as Positive to the total number of Positive samples**. The recall measures the model's ability to detect positive samples. The higher the recall, the more positive samples detected.

$$Recall = \frac{True\ positive}{True\ positive + False\ negative}$$

F1 Score

F1-score is **one of the most important evaluation metrics in machine learning**. It elegantly sums up the predictive performance of a model by combining two otherwise competing metrics – precision and recall.

$$F1 = 2 \cdot \frac{(Precision * Recall)}{Precision + Recall}$$

AUC - ROC

The Receiver Operator Characteristic (ROC) is a probability curve that plots the TPR(True Positive Rate) against the FPR(False Positive Rate) at various threshold values and separates the 'signal' from the 'noise'. The **Area Under the Curve (AUC)** is the measure of the ability of a classifier to distinguish between classes.

Summary

- The output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc.
- In Classification, a program learns from the given dataset or observations and then classifies new observation into several classes or groups.
- Classes can be called as targets/labels or categories.
- Since the Classification algorithm is a supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output.
- Linear models of classification are logistic regression and SVM.
- Nonlinear models are KNN, Kernel SVM, Naïve Bayes, Decision tree, Random Forest.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- Naïve Bayes performs well in multi-class predictions as compared to the other Algorithms.
- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Keywords

- **Classification:** It is the operation of separating various entities into several classes.
- **Classification algorithm:** The Classification algorithm is a supervised learning technique that is used to identify the category of new observations on the basis of training data.
- **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier.
- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.
- **Lazy Learners:** Lazy Learner firstly stores the training dataset and wait until it receives the test dataset. In Lazy learner case, classification is done on the basis of the most related data stored in the training dataset. It takes less time in training but more time for predictions.
- **Eager Learners:** Eager Learners develop a classification model based on a training dataset before receiving a test dataset. Opposite to Lazy learners, Eager Learner takes more time in learning, and less time in prediction.
- **Feature:** It is an individual measurable property of a phenomenon being observed.
- **Naïve Bayes:** It is a supervised learning algorithm. It is based on Bayes theorem. It is mainly used in text classification that includes a high-dimensional training dataset.
- **Cross validation:** Cross-validation is one of the most widely used data resampling methods to assess the generalization ability of a predictive model
- **AUC-ROC:** The Receiver Operator Characteristic (ROC) is a probability curve that plots the TPR(True Positive Rate) against the FPR(False Positive Rate) at various threshold values and separates the 'signal' from the 'noise'. The **Area Under the Curve (AUC)** is the measure of the ability of a classifier to distinguish between classes.

SelfAssessment

1. The supervised machine learning algorithms are
 - A. Classification algorithms
 - B. Regression algorithms
 - C. Both of the above
 - D. None of the above

2. Which learner firstly stores the training dataset and wait until it receives the test dataset?
 - A. Lazy learner
 - B. Eager learner
 - C. Either of the above
 - D. None of the above

3. Which learners develop a classification model based on a training dataset before receiving a test dataset?
 - A. Lazy learner
 - B. Eager learner
 - C. Either of the above
 - D. None of the above

4. Eager Learner takes time in learning, and time in prediction.
 - A. More, less
 - B. Less, more
 - C. Less, less
 - D. More, more

5. Lazy learners take time in training but time for predictions
 - A. More, less
 - B. Less, more
 - C. Less, less
 - D. More, more

6. Which of the following is not a nonlinear model for classification?
 - A. KNN
 - B. Kernel SVM
 - C. Logistic Regression
 - D. Decision Tree

7. K-NN is a, which means it does not make any assumption on underlying data.
 - A. Parametric
 - B. Non-parametric
 - C. Functional

- D. None of the above
8. KNN is a
- A. Eager learner
 - B. Lazy learner
 - C. Not a learner
 - D. None of the above
9. Which of the following is the advantage of KNN algorithm?
- A. Simple to implement
 - B. Robust to noisy training data
 - C. Effective if the training data is large
 - D. All of the above
10. Which of the following is the disadvantage of KNN algorithm?
- A. Determining the value of k in advance
 - B. High computational cost
 - C. Both of the above
 - D. None of the above
11. The Naïve Bayes algorithm
- A. Is a supervised learning algorithm
 - B. Is based upon Bayes theorem
 - C. Mainly used for text classifications
 - D. All of the above
12. $P(A|B)$ in Bayes theorem is
- A. Posterior probability
 - B. Prior probability
 - C. Likelihood probability
 - D. Marginal probability
13. What is marginal probability?
- A. Probability of evidence
 - B. Probability of hypothesis
 - C. Both of the above
 - D. None of the above
14. Naïve Bayes works for
- A. Binary classes
 - B. Multi-classes
 - C. Both of the above

D. None of the above

15. Naïve Bayes algorithm assumes that all features are

- A. Independent
- B. Unrelated
- C. Both of the above
- D. None of the above

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. C | 2. A | 3. B | 4. A | 5. B |
| 6. C | 7. B | 8. B | 9. D | 10. C |
| 11. D | 12. A | 13. A | 14. C | 15. C |

Review Questions

1. What is supervised learning? Explain its types and give few examples.
2. What is classification and classification algorithm? What are the types of classification?
3. State the difference between classification and regression.
4. What is learning in classification problems? Explain its types.
5. What are linear and non-linear models in classification algorithms. Give examples of both.
6. What are the applications of classification algorithms? Tell any 5 in detail.
7. What is K-NN algorithm? What is its need? Also explain its working.
8. In KNN algorithm, how do we select the value of K? What are the advantages and disadvantages of KNN algorithm?
9. What is Naïve Bayes algorithm? Also explain the Bayes theorem.
10. What are the steps of Naïve Bayes algorithm? Explain it with an example.
11. What are the advantages, disadvantages, and applications of Naïve Bayes algorithm?
12. What is cross validation? Explain its types.
13. What is the performance metrics of classification algorithms? Explain.



Further Readings

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>



Web Links

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Unit 09: Regression Models

CONTENTS

Objectives

Introduction

9.1 Regression

9.2 Machine Linear Regression

9.3 Machine Logistic Regression

9.4 Regularization

9.5 Performance Metric of Regression

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this unit, you will be able to:

- Understand the meaning of regression
- Know the types of regression
- Understand machine linear regression and machine logistic regression
- Understand the concept of regularization
- Understand the performance metrics of regression

Introduction

Regression is also one of the supervised learning techniques. It is different from the concept of classification. It is a task of predicting the continuous quantity. Regression is known as a process of finding a model that predicts a continuous value based on its input variables.

9.1 Regression

In regression problems, the goal is to mathematically estimate a mapping function (f) from the input variables (x) to the output variables (y).



Example:

- Consider a dataset that contains information about all the students in a university. An example of a regression task would be to predict the height of any student based on their gender, weight, major, and diet. We can do this because height is a continuous quantity, i.e., there are an infinite number of possible values for a person's height.

Formal definition of regression

Regression analysis is a way of predicting future happenings between a dependent (target) and one or more independent variables (also known as a predictor).



Example 1: To predict the relationship between reckless driving and the total number of road accidents caused by a driver.



Example 2: The effect on sales and spending a certain amount of money on advertising.

Regression vs Classification

Undoubtedly, both regression and classification are supervised learning techniques. But still there is a major difference in classification and regression.

Regression	Classification
Predicting the continuous quantity.	Predicting the categorical value

Applicability

The concept of regression is applied to many areas for solving different kinds of problems. These are:

- Financial forecasting (like house price estimates, or stock prices)
- Sales and promotions forecasting
- Testing automobiles
- Weather analysis and prediction
- Time series forecasting

Related terms

- **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called target variable.
- **Independent Variable:** The factors which affect the dependent variables, or which are used to predict the values of the dependent variables are called independent variable, also called as a predictor.
- **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.
- **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because it creates problem while ranking the most affecting variable.
- **Underfitting and Overfitting:** If our algorithm works well with the training dataset but not well with test dataset, then such problem is called Overfitting. And if our algorithm does not perform well even with training dataset, then such problem is called underfitting.

Reasons of using regression analysis

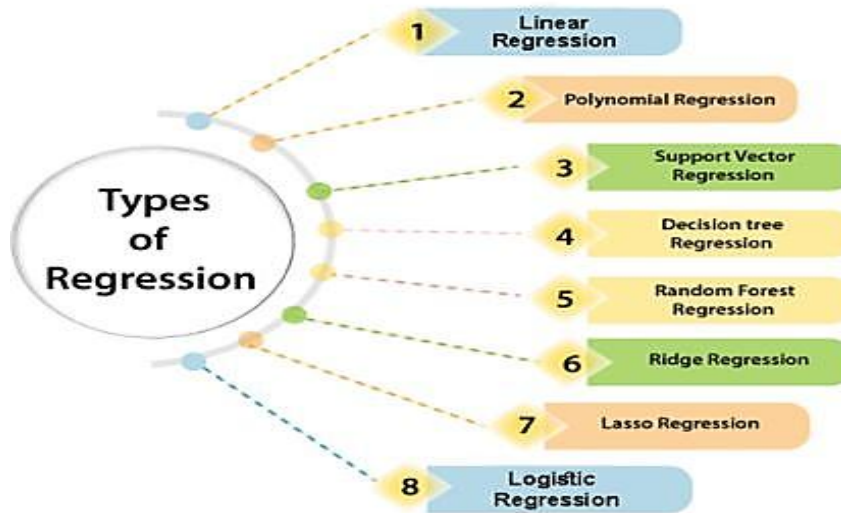
Regression estimates the relationship between the target and the independent variable. It is used to find the trends in data. It helps to predict real/continuous values. By performing the regression, we can confidently determine the most important factor, the least important factor, and how each factor is affecting the other factors.

Types of Regression

There are various types of regression. These are:

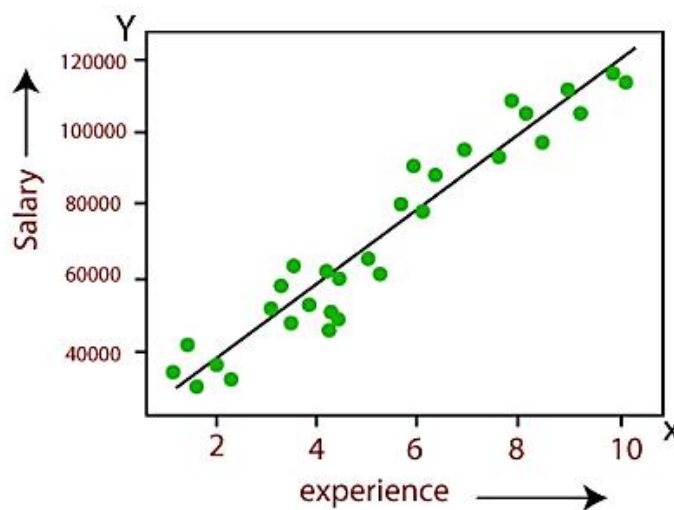
1. Linear regression

2. Polynomial regression
3. Support vector regression
4. Decision tree regression
5. Random forest regression
6. Lasso regression
7. Logistic regression



9.2 Machine Linear Regression

Linear regression is a statistical regression method which is used for predictive analysis. It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables. It is used for solving the regression problem in machine learning. Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression. If there is only one input variable (x), then such linear regression is called simple linear regression. And if there is more than one input variable, then such linear regression is called multiple linear regression. The relationship between variables in the linear regression model can be explained using the below image. Here we are predicting the salary of an employee based on the year of experience.



Mathematical equation for Linear regression: $Y = aX + b$

Here, Y = dependent variables (target variables), X = Independent variables (predictor variables), a and b are the linear coefficients

Applications of linear regression

Some popular applications of linear regression are:

- Analyzing trends and sales estimates
- Salary forecasting
- Real estate prediction
- Arriving at ETAs in traffic.

9.3 Machine Logistic Regression

Logistic regression is a type of regression, but it is different from the linear regression algorithm in the term how they are used. Logistic regression is another supervised learning algorithm which is used to solve the classification problems. In **classification problems**, we have dependent variables in a binary or discrete format such as 0 or 1. Logistic regression algorithm works with the categorical variable such as 0 or 1, Yes or No, True or False, Spam or not spam, etc. It is a predictive analysis algorithm which works on the concept of probability.

Use of Function in Machine Logistic Regression

Logistic regression uses **sigmoid function** or logistic function which is a complex cost function. This sigmoid function is used to model the data in logistic regression. The function can be represented as: $f(x) = 1/(1 + e^{-x})$

- $f(x)$ = Output between the 0 and 1 value.
- x = input to the function
- e = base of natural logarithm.

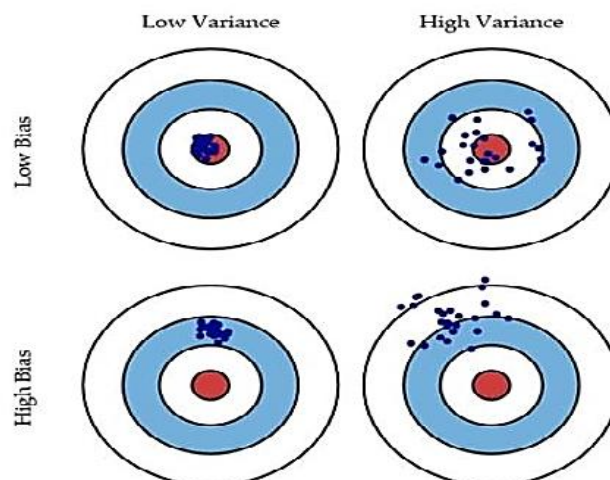
Types of machine logistic regression

There are three types of logistic regression:

- Binary(0/1, pass/fail)
- Multi(cats, dogs, lions)
- Ordinal(low, medium, high)

Bias and variance in regression models

- Bias is the simplifying assumptions made by a model to make the target function easier to learn.
- Variance is the amount that the estimate of the target function will change if different training data was used.



1. Very Accurate Model – therefore the error of our model will be low, meaning a **low bias and low variance** as shown in the first part of the figure.
2. As variance increases, the spread of our data point increases which result in less accurate prediction.
3. As Bias increases the error between our predicted value and the observed values increases. A high bias assumes a strong assumption or strong restrictions on the model.

There are two major issues which comes under consideration are:

- **Under fitting** – under fitting model performs poorly on training data. This happens because the model is unable to capture the relationship between the input example and the target variable. To overcome under fitting or high bias, we can basically add new parameters to our model so that the model complexity increases, and thus reducing high bias.
- **Over fitting** – As we add more and more parameters to our model, its complexity increases, which results in increasing variance and decreasing bias? To overcome over fitting there are two ways –
 1. Reduce the model complexity
 2. Regularization

9.4 Regularization

Regularization Techniques is an unavoidable and important step to improve the model prediction and reduce errors. One of the major aspects of training your machine learning model is avoiding over fitting. The model will have a low accuracy if it is over fitting. This happens because your model is trying too hard to capture the noise in your training dataset. By noise we mean the data points that don't really represent the true properties of your data, but random chance. Learning such data points, makes your model more flexible, at the risk of over fitting. Regularization is a technique that helps in avoiding over fitting and increasing model interpretability. This is a form of regression that constrains / regularizes or shrinks the coefficient estimates towards zero. In other words, this technique discourages learning a more complex or flexible model, to avoid the risk of over fitting.

9.5 Performance Metric of Regression

There are various performance metrics of regression. These are

- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Root Mean Squared Error (RMSE)
- R-Squared
- Adjusted R-squared

Mean Absolute Error

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - y_i'|$$

y_i = Actual expected output

\hat{y}_i = Model's prediction

Mean Squared Error

$$MSE = \frac{1}{N} \sum_{i=1}^n (y_i - y_i')^2$$

Here, the error term is squared and thus more sensitive to outliers as compared to Mean Absolute Error (MAE).

Root Mean Squared Error

$$RMSE = \frac{\sqrt{1}}{n} \sum_{j=1}^n (y_j - y'_j)^2$$

MSE includes squared error terms, we take the square root of MSE which gives the RMSE

R-Squared

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - y'_i)^2}{\sum_i (y_i - \bar{y})^2}$$

R-squared is also known as the **Coefficient of Determination**. It explains the degree to which the input variables explain the variation of the output / predicted variable.

Adjusted R Square

$$AdjustedR^2 = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

Here, N- total sample size (number of rows) and p- number of predictors (number of columns). The **limitation of R-squared** is that it will *either stay the same or increases with the addition of more variables, even if they do not have any relationship with the output variables.

Summary

- Regression is a task of predicting the continuous quantity. It is the process of finding a model that predicts a continuous value based on its input variables.
- In regression problems, the goal is to mathematically estimate a mapping function (f) from the input variables (x) to the output variables (y).
- If our algorithm works well with the training dataset but not well with test dataset, then such problem is called over fitting. And if our algorithm does not perform well even with training dataset, then such problem is called under fitting.
- Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
- If there is only one input variable (x), then such linear regression is called simple linear regression. And if there is more than one input variable, then such linear regression is called multiple linear regression.
- Logistic regression uses sigmoid function or logistic function which is a complex cost function.
- To overcome over fitting there are two ways – reduce the model complexity and Regularization
- The **limitation of R-squared** is that it will *either stay the same or increases with the addition of more variables, even if they do not have any relationship with the output variables.

Keywords

- **Regression:** It is the process of finding a model that predicts a continuous value based on its input variables.
- **Regression analysis:** It is a way of predicting future happenings between a dependent (target) and one or more independent variables (also known as a predictor).
- **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called target variable.

- **Independent Variable:** The factors which affect the dependent variables, or which are used to predict the values of the dependent variables are called independent variable, also called as a predictor.
- **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.
- **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because it creates problem while ranking the most affecting variable.
- **Linear Regression:** Linear regression is a statistical regression method which is used for predictive analysis.
- **Logistic regression:** It is another supervised learning algorithm which is used to solve the classification problems. In **classification problems**, we have dependent variables in a binary or discrete format such as 0 or 1.
- **Bias:** It is the simplifying assumptions made by a model to make the target function easier to learn.
- **Variance:** variance is the amount that the estimate of the target function will change if different training data was used.
- **Under fitting** – under fitting model performs poorly on training data. This happens because the model is unable to capture the relationship between the input example and the target variable.
- **Over fitting** – As we add more and more parameters to our model, its complexity increases, which results in increasing variance and decreasing bias.

SelfAssessment

1. Regression is the task of predicting
 - A. Continuous quantities
 - B. Categorical values
 - C. Both of these
 - D. None of the above

2. Using supervised machine learning techniques, we can
 - A. Predict the continuous quantities
 - B. Predict the categorical values
 - C. Both of the above
 - D. None of the above

3. Which of the following are the applications of regression?
 - A. Sales and promotion forecasting
 - B. Testing automobiles
 - C. Time series forecasting
 - D. All of the above

4. The factor which we want to predict or understand in regression is
 - A. Dependent variable
 - B. Independent variable
 - C. Non-dependent variable

- D. None of the above
5. The factors which are used to predict the values are known as
- A. Dependent variable
 - B. Independent variables
 - C. Pending variables
 - D. None of the above
6. What is an outlier?
- A. A value much lesser than other values
 - B. A value much greater than other values
 - C. Either of the above
 - D. None of the above
7. The concept of multicollinearity should be avoided for ranking the most affecting variable.
- A. True
 - B. False
8. What kind of issue occurs if the algorithm does not even work well with training data?
- A. Overfitting
 - B. Underfitting
 - C. Any of these
 - D. None of the above
9. If an algorithm works well with training data, but not with testing data. Then what kind of issue can occur?
- A. Overfitting
 - B. Underfitting
 - C. Any of these
 - D. None of the above
10. By performing regression, we can determine
- A. Most important factor
 - B. Least important factor
 - C. Reason of one factor affecting the another
 - D. All of the above
11. Which of these are the types of regression?
- A. Lasso regression
 - B. Decision tree regression
 - C. Support vector regression
 - D. All of the above

12. Which of the following is used to solve the classification problem?
 - A. K-means clustering
 - B. Linear regression
 - C. Logistic regression
 - D. Hierarchical clustering

13. Which of the following are types of logistic regression?
 - A. Binary
 - B. Multi
 - C. Ordinal
 - D. All of the above

14. Why is regularization techniques used?
 - A. To reduce error
 - B. To improve the model prediction
 - C. To make a good model
 - D. All of the above

15. The performance metrics of regression are
 - A. Mean absolute error
 - B. Mean Squared error
 - C. Root mean squared error
 - D. All of the above

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. A | 2. C | 3. D | 4. A | 5. B |
| 6. C | 7. A | 8. B | 9. A | 10. D |
| 11. D | 12. C | 13. D | 14. D | 15. D |

Review Questions

1. What is regression? Give its formal definition. Also tell how it is different from classification?
2. What is the goal of regression in machine learning? Also tell what are the applications of regression?
3. What are the types of regression? Explain linear regression and logistic regression.
4. What is machine linear regression? Also give few applications of it.
5. What is machine logistic regression? Also give the use of function in it. Explain its types as well.
6. Explain the performance metrics of regression in detail.



Further Readings

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>



Web Links

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Chapter 10: Weka

CONTENTS

Objectives

Introduction

10.1 WEKA

10.2 Download Weka

10.3 GUI Selector

10.4 Clustering of Data

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this unit, you will be able to

- Understand Weka tool
- Understand how to import data in Weka
- Understand how to cluster and classify the data in Weka

Introduction

WEKA was developed at the University of Waikato in New Zealand; the name stands for Waikato Environment for Knowledge Analysis.

10.1 WEKA

It has a great collection of machine learning algorithms and data preprocessing tools. It provides extensive support for the whole process of experimental data mining, including preparing the input data, evaluating learning schemes statistically, and visualizing the input data and the result of learning. It also includes a variety of tools for transforming datasets, such as the algorithms for discretization and sampling. You can preprocess a dataset, feed it into a learning scheme, and analyze the resulting classifier and its performance. The workbench includes methods for the main data mining problems: regression, classification, clustering, association rule mining, and attribute selection. All algorithms take their input in the form of a single relational table that can be read from a file or generated by a database query. One way of using WEKA is to apply a learning method to a dataset and analyze its output to learn more about the data. Another is to use learned models to generate predictions on new instances. A third is to apply several different learners and compare their performance in order to choose one for prediction.

How to use?

- The easiest way to use WEKA is through a graphical user interface called the Explorer.
- The Knowledge Flow interface allows you to design configurations for streamed data processing.

- WEKA's third interface, the Experimenter, is designed to help you answer a basic practical question when applying classification and regression techniques: Which methods and parameter values work best for the given problem?
- The fourth interface, called the Workbench, is a unified graphical user interface that combines the other three (and any plugins that the user has installed) into one application.

10.2 Download Weka

WEKA is available from <http://www.cs.waikato.ac.nz/ml/weka>. When you open this website, click on softwares. Under softwares, click on Download tab. It will ask you to choose the operating system of your computer. So, choose the appropriate option of operating system and download it. After downloading, double click on .exe file to start installation. Then click next. Then agree to the terms of service by checking the box. Then after that select the components that you want to install and unselect the components that you want to uninstall. After that it will ask you to choose the installation location and start folder. The installation will begin, and it will be completed in few seconds.

[Project](#) [Software](#) [Book](#) [Courses](#) [Publications](#) [People](#) [Related](#)

Weka 3: Machine Learning Software in Java

Weka is a collection of machine learning algorithms for data mining tasks. It contains tools for data preparation, classification, regression, clustering, association rules mining, and visualization.

Found only on the islands of New Zealand, the Weka is a flightless bird with an inquisitive nature. The name is pronounced like [this](#), and the bird sounds like [this](#).

Weka is open source software issued under the **GNU General Public License**.

We have put together several **free online courses** that teach machine learning and data mining using Weka. The videos for the courses are available on **Youtube**.

Weka supports **deep learning**!

Getting started

- **Requirements**
- **Download**
- **Documentation**
- **FAQ**
- **Getting Help**

Further information

- **Citing Weka**
- **Datasets**
- **Related Projects**
- **Miscellaneous Code**
- **Other Literature**

Developers

- **Development**
- **History**
- **Subversion**
- **Contributors**
- **Commercial licenses**

Home

Downloading and installing Weka

Snapshots

Stable version

Windows

Mac OS - Intel processors

Mac OS - ARM processors

Linux

Other platforms

Developer version

Windows

Mac OS - Intel processors

Mac OS - ARM processors

Linux

Other platforms

Old versions

Upgrading from Weka 3.7

- Click here to download a self-extracting executable for 64-bit Windows that includes Azul's 64-bit OpenJDK Java VM 17 (`weka-3-8-6-azul-zulu-windows.exe`; 133.2 MB)

This executable will install Weka in your Program Menu. Launching via the Program Menu or shortcuts will automatically use the included JVM to run Weka.

Mac OS - Intel processors

- Click here to download a disk image for Mac OS that contains a Mac application including Azul's 64-bit OpenJDK Java VM 17 for Intel Macs. (`weka-3-8-6-azul-zulu-osx.dmg`; 180.2 MB)

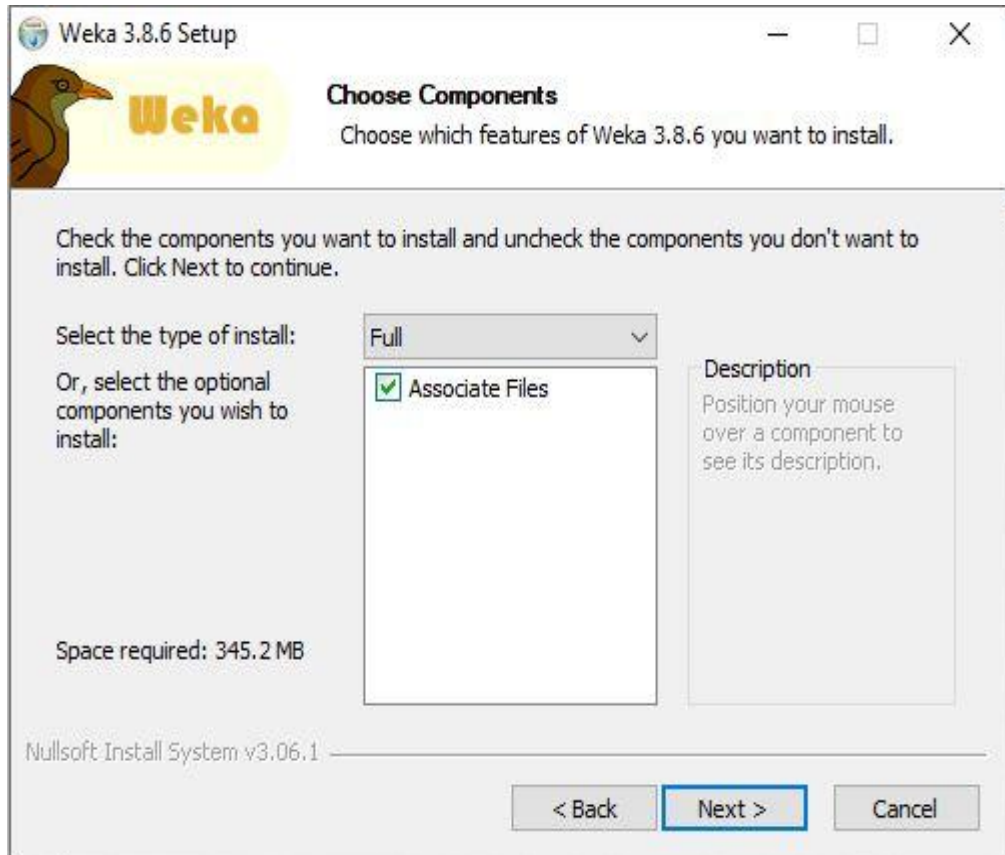
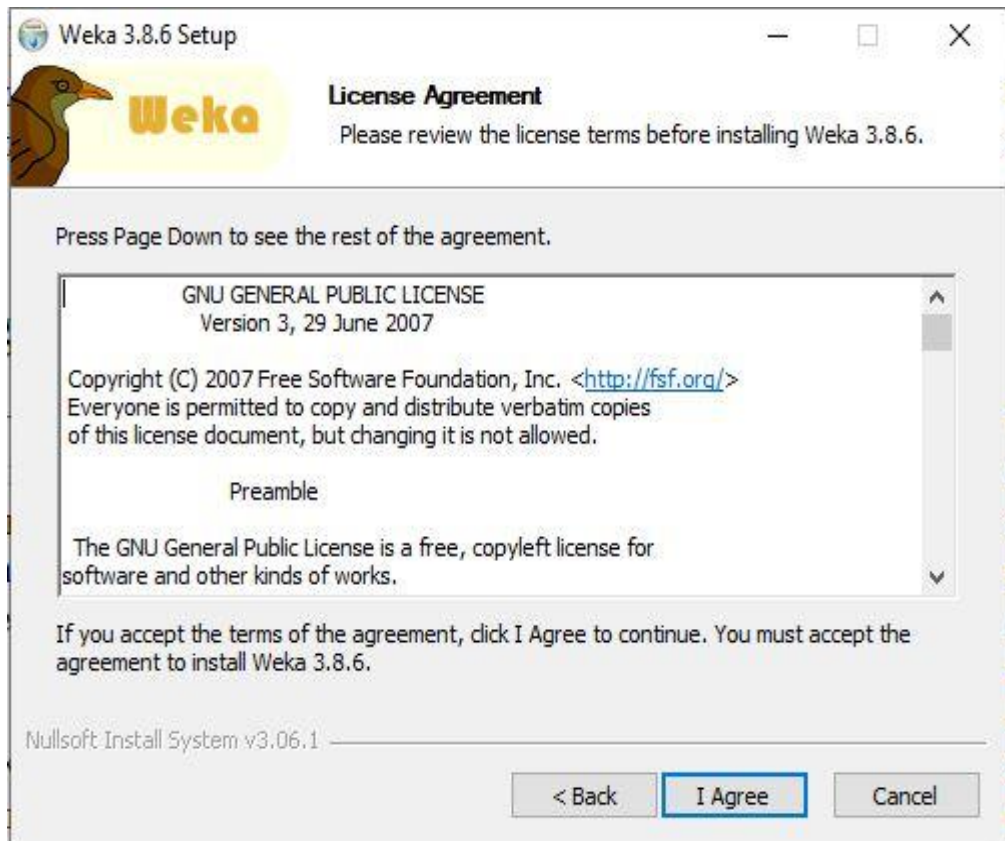
Mac OS - ARM processors

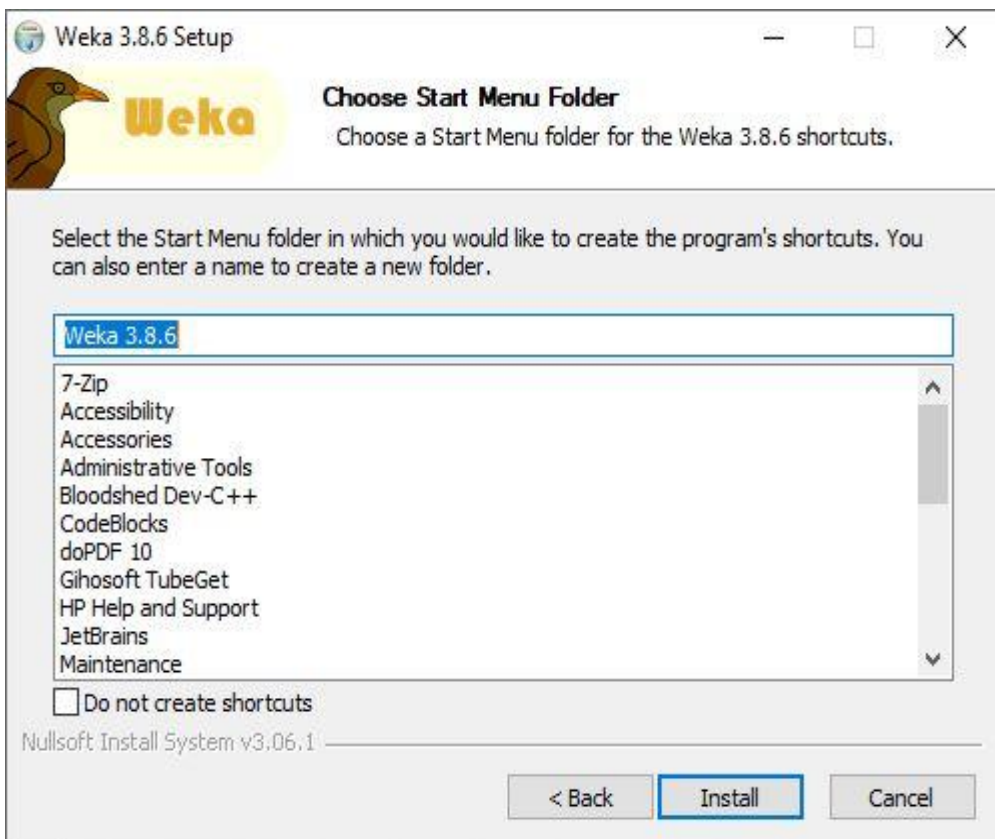
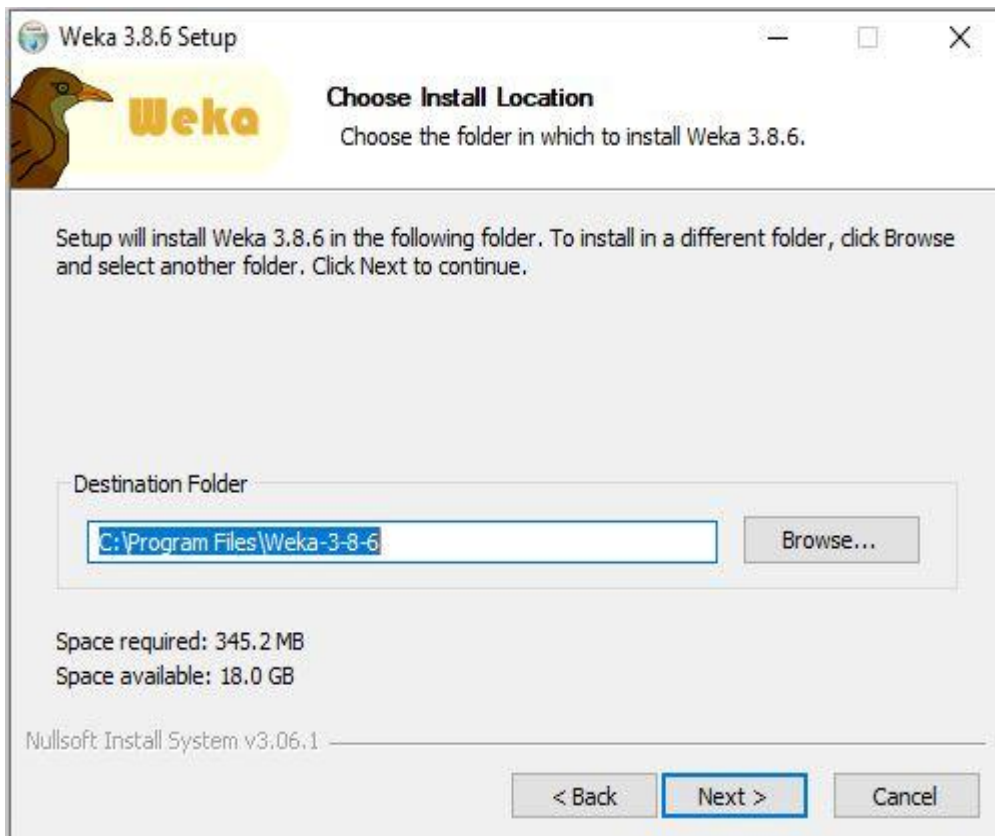
- Click here to download a disk image for Mac OS that contains a Mac application including Azul's 64-bit OpenJDK Java VM 17 for ARM Macs. (`weka-3-8-6-azul-zulu-arm-osx.dmg`; 166.3 MB)

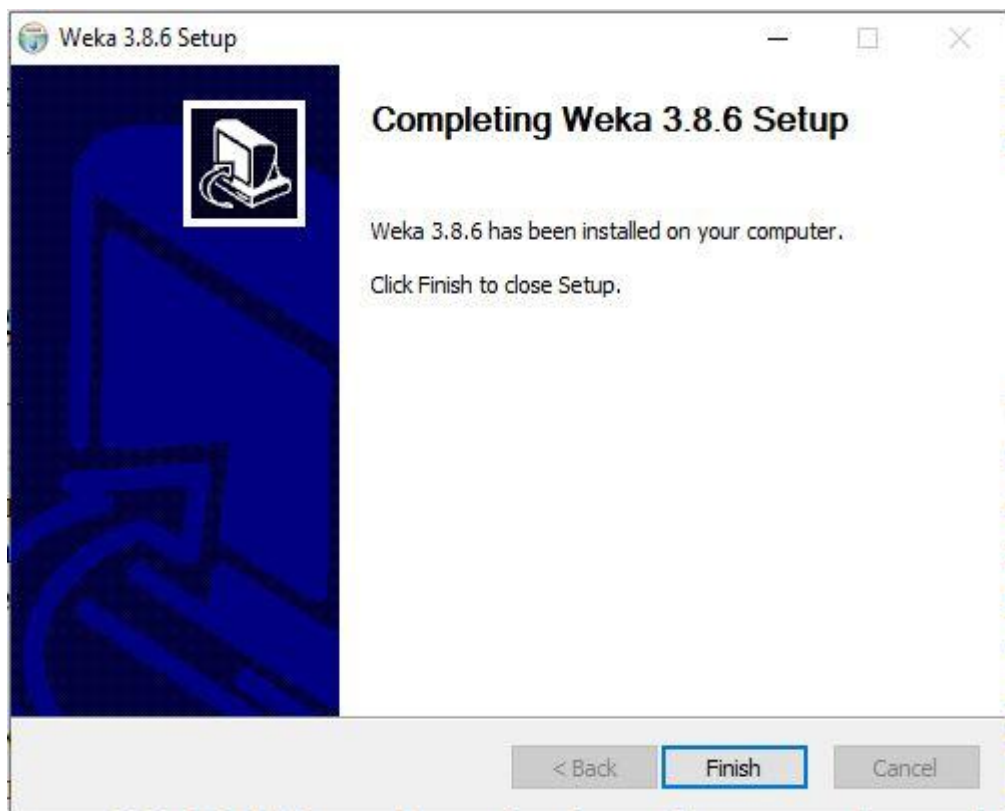
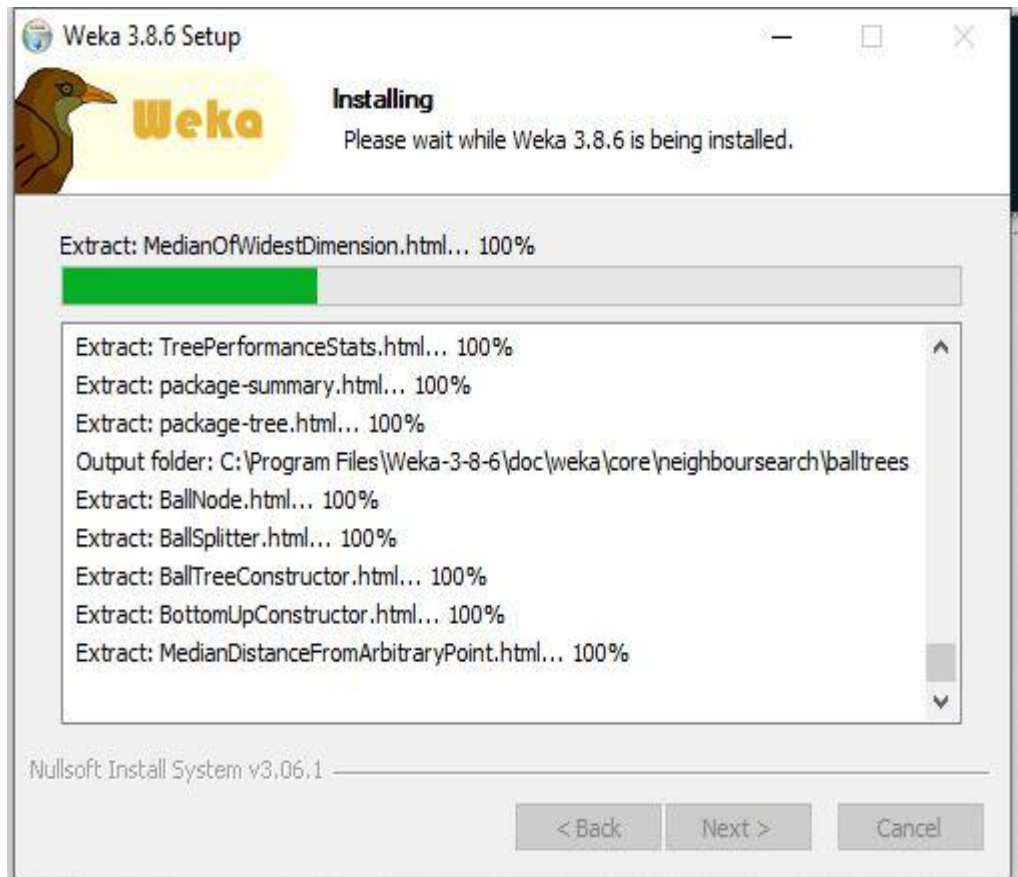
Linux

- Click here to download a zip archive for Linux that includes Azul's 64-bit OpenJDK Java VM 17 (`weka-3-8-6-azul-zulu-linux.zip`; 146.9 MB)









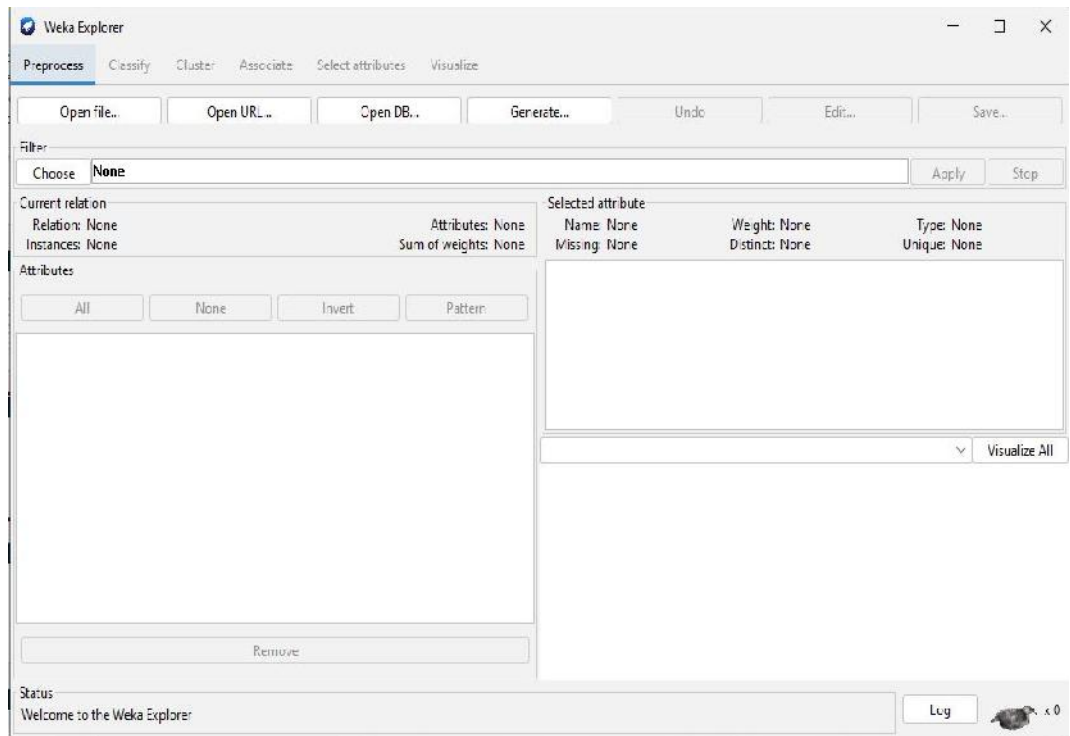
10.3 GUI Selector

When the Weka is installed, the first screen that will be visible to you is GUI selector. There are a total of 5 applications available. These are: Explorer, Experimenter, Knowledge Flow, Workbench and Simple CLI. Click on any application to start with it.



This image below shows the Explorer application. Under this, we have various panels. These are

- Pre-process:
- Classify
- Cluster
- Associate
- Select Attributes
- Visualize



Preparing the Data

The data is often presented in a spreadsheet or database. However, WEKA's native data storage method is ARFF format. You can easily convert from a spreadsheet to ARFF. The bulk of an ARFF file consists of a list of the instances, and the attribute values for each instance are separated by commas. Most spreadsheet and database programs allow you to export data into a file in comma-separated value (CSV) format as a list of records with commas between items.

Load the data

Click the Open file button to bring up a standard dialog through which you can select a file. Choose the file. If you have it in CSV format, change from *ARFF data files* to *CSV data files*. When you specify a .csv file it is automatically converted into ARFF format.

Building a decision tree model

Click the *Classify* tab to get a screen for classification. First select the classifier by clicking the *Choose* button at the top left, opening the trees section of the hierarchical menu and finding *J48*.

Estimate the model

See the confusion matrix to find out the results.

Unit 10: Weka

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

Choose J48 - C.0.25 - M 2

Test options

Use training set

Supplied test set Set...

Cross-validation Folds 10

Percentage split % 50

More options...

(None) Type

Start Stop

Result list (right-click for options)

J48: 9 - trees J48

Classifier output

=== Run information ===

Schema: weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: Glass

Instances: 214

Attributes: 10

EC

Na

Mg

AL

SI

K

Ca

Fe

Te

Type

Test mode: 10-fold cross-validation

J48 pruned tree

```

-----
Ba <= 0.27
| Mg <= 2.41
| | K <= 0.03
| | | Na <= 13.75: build wind non-float (3.0)
| | | Na > 13.75: tableware (9.0)
| | | K > 0.03
| | | | Na <= 13.49
| | | | | RI <= 1.5241: containers (13.0/1.0)
| | | | | RI > 1.5241: build wind non-float (3.0)
| | | | Na > 13.49: build wind non-float (7.0/1.0)
| | Mg > 2.41
| | | Al <= 1.41
| | | | RI <= 1.51707
| | | | | RI <= 1.51596: build wind float (3.0)
| | | | | RI > 1.51596
| | | | | | Fe <= 0.12
| | | | | | Mg <= 3.54: vehic wind float (5.0)
| | | | | | Mg > 3.54
| | | | | | | RI <= 1.51667: build wind non-float (2.0)
| | | | | | | RI > 1.51667: vehic wind float (2.0)
| | | | | | | Fe > 0.12: build wind non-float (2.0)
| | | | | RI > 1.51707
| | | | | | K <= 0.23
| | | | | | Mg <= 3.34: build wind non-float (2.0)
| | | | | | Mg > 3.34
| | | | | | | Si <= 72.64
| | | | | | | Na <= 14.01: build wind float (14.0)

```

Data Science Toolbox

```

Number of Leaves :    30
Size of the tree :    59

Time taken to build model: 0.06 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      143          66.8224 %
Incorrectly Classified Instances     71          33.1776 %
Kappa statistic                     0.55
Mean absolute error                  0.1026
Root mean squared error              0.2897
Relative absolute error              48.4507 %
Root relative squared error         89.2727 %
Total Number of Instances           214

=== Detailed Accuracy By Class ===

      TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
0.714  0.174  0.667    0.714  0.690    0.532  0.806    0.667    build wind float
0.618  0.181  0.653    0.618  0.635    0.443  0.768    0.606    build wind non-float
0.353  0.046  0.400    0.353  0.375    0.325  0.766    0.251    vehic wind float
?      0.000  ?        ?        ?        ?      ?      ?      vehic wind non-float
0.769  0.010  0.833    0.769  0.800    0.788  0.872    0.575    containers
0.778  0.029  0.538    0.778  0.636    0.629  0.930    0.527    tableware
0.793  0.022  0.852    0.793  0.821    0.795  0.869    0.738    headlamps
Weighted Avg.  0.668  0.130  0.670    0.668  0.668  0.539  0.807    0.611

```

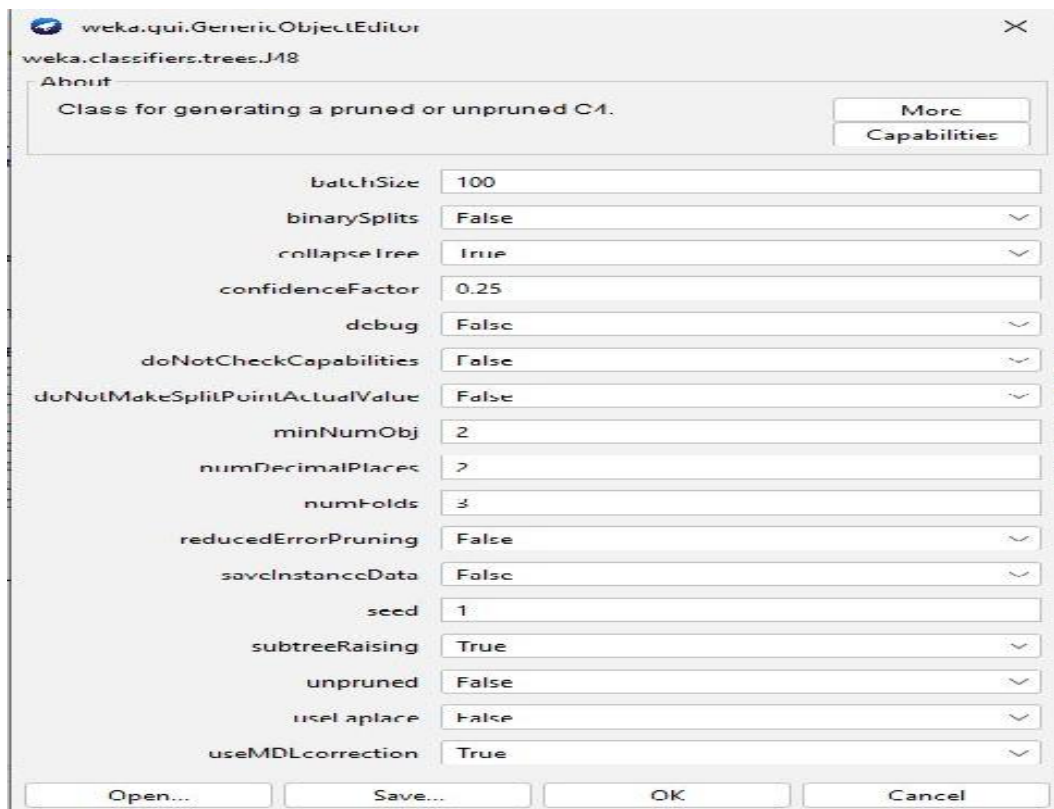
=== Confusion Matrix ===

```

  a  b  c  d  e  f  g  <-- classified as
50 15  3  0  0  1  1 | a = build wind float
16 47  6  0  2  3  2 | b = build wind non-float
 5  5  6  0  0  1  0 | c = vehic wind float
 0  0  0  0  0  0  0 | d = vehic wind non-float
 0  2  0  0 10  0  1 | e = containers
 1  1  0  0  0  7  0 | f = tableware
 3  2  0  0  0  1 23 | g = headlamps

```

The image shown below is the configuration panel of classifier which we have implemented. We can change the value of any parameter here. For example: unpruned \rightarrow False to true or we can change the batch size as well. Once the parameters are changed, we can see the results using the confusion matrix.



The result after changing the configuration

Correctly Classified Instances	144	67.2897 %
Incorrectly Classified Instances	70	32.7103 %
Kappa statistic	0.5571	
Mean absolute error	0.1001	
Root mean squared error	0.2854	
Relative absolute error	47.2665 %	
Root relative squared error	87.9232 %	
Total Number of Instances	214	

Visualization of tree

The Visualize panel helps you visualize a dataset—not the result of a classification or clustering model, but the dataset itself. It displays a matrix of two-dimensional scatter plots of every pair of attributes.

Use of filter

These are accessible from the Explorer, and also from the Knowledge Flow and Experimenter interfaces. All filters transform the input dataset in some way. When a filter is selected using the Choose button, its name appears in the line beside that button. Click that line to get a generic object editor to specify its properties. What appears in the line is the command-line version of the filter, and the parameters are specified with minus signs. This is a good way of learning how to use the WEKA commands directly. There are two kinds of filter: unsupervised and supervised. This

seemingly innocuous distinction masks a rather fundamental issue. Filters are often applied to a training dataset and then also applied to the test file. If the filter is supervised—for example, if it uses class values to derive good intervals for discretization—applying it to the test data will bias the results. It is the discretization intervals derived from the training data that must be applied to the test data.

10.4 Clustering of Data

Use the Cluster and Associate panels to invoke clustering algorithms and methods for finding association rules. When clustering, WEKA shows the number of clusters and how many instances each cluster contains. For some algorithms the number of clusters can be specified by setting a parameter in the object editor. For probabilistic clustering methods, WEKA measures the log-likelihood of the clusters on the training data: the larger this quantity, the better the model fits the data. Increasing the number of clusters normally increases the likelihood but may overfit. The controls on the Cluster panel are like those for Classify. You can specify some of the same evaluation methods—use training set, supplied test set, and percentage split (the last two are used with the log-likelihood). A further method, classes to clusters evaluation, compares how well the chosen clusters match a preassigned class in the data. You select an attribute (which must be nominal) that represents the “true” class. Having clustered the data, WEKA determines the majority class in each cluster and prints a confusion matrix showing how many errors there would be if the clusters were used instead of the true class. If your dataset has a class attribute, you can ignore it during clustering by selecting it from a pull-down list of attributes and see how well the clusters correspond to actual class values. Finally, you can choose whether to store the clusters for visualization.

Summary

- WEKA was developed at the University of Waikato in New Zealand; the name stands for Waikato Environment for Knowledge Analysis.
- It also includes a variety of tools for transforming datasets, such as the algorithms for discretization and sampling.
- The workbench includes methods for the main data mining problems: regression, classification, clustering, association rule mining, and attribute selection.
- One way of using WEKA is to apply a learning method to a dataset and analyze its output to learn more about the data. Another is to use learned models to generate predictions on new instances. A third is to apply several different learners and compare their performance in order to choose one for prediction.
- The panels in Explorer are Pre-process, classify, cluster, associate, select Attributes and visualize.

Keywords

- **WEKA:** It is a collection of machine learning algorithms and data preprocessing tools.
- **Knowledge Flow:** The Knowledge Flow interface allows you to design configurations for streamed data processing.
- **Workbench:** It is a unified graphical user interface that combines the other three (and any plugins that the user has installed) into one application.
- **Clustering:** It is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.
- **Classification:** The definition of classifying is categorizing something or someone into a certain group or system based on certain characteristics. An example of classifying is assigning plants or animals into a kingdom and species.

SelfAssessment

1. What is WEKA?
 - A. Waikato Environment for Known Analysis
 - B. Waikato Environment for Knowledge Analysis
 - C. Waste Environment for Knowledge Analysis
 - D. None of the above

2. Weka was developed in?
 - A. New Zealand
 - B. New York
 - C. India
 - D. None of the above

3. Weka has a great collection of
 - A. Data mining tools
 - B. Clustering algorithm
 - C. Classification algorithms
 - D. All of the above

4. With Weka, we can
 - A. Preprocess a dataset
 - B. Feed it into a learning scheme
 - C. Analyze the classifier and corresponding performance
 - D. All of the above

5. The easiest interface of WEKA is
 - A. Explorer
 - B. Knowledge Flow
 - C. Experimenter
 - D. Workbench

6. Which of the following GUI combines the other three applications?
 - A. Explorer
 - B. Knowledge Flow
 - C. Experimenter
 - D. Workbench

7. Under Weka Explorer which tabs are available?
 - A. Pre-process
 - B. Classify
 - C. Cluster
 - D. All of the above

8. Which filters are available in Weka?
 - A. Supervised
 - B. Unsupervised
 - C. Both of the above
 - D. None of the above

9. Which of the following algorithms are available for classification in Weka?
 - A. Decision Tree
 - B. Random Forest
 - C. Naïve Bayes
 - D. All of the above

10. Under cluster tab, which options are available?
 - A. Simple K means
 - B. Filtered clusterer
 - C. Hierarchical cluster
 - D. All of the above

11. In loading data in Weka, the data can be loaded from
 - A. Local file system
 - B. Web
 - C. Database
 - D. Either of the above

12. Which of the following file formats are supported in Weka?
 - A. csv
 - B. arff
 - C. json
 - D. All of the above

13. Which of the following is used for creating association rules?
 - A. Naïve Bayes
 - B. Hierarchical clustering
 - C. Random forest
 - D. Apriori algorithm

14. Which of the following can be performed in Weka?
 - A. Associate
 - B. Attribute selection
 - C. Cluster
 - D. All of the above

15. Weka was developed in?

- A. New Zealand
- B. New York
- C. India
- D. None of the above

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. B | 2. A | 3. D | 4. D | 5. A |
| 6. D | 7. D | 8. C | 9. D | 10. D |
| 11. D | 12. D | 13. D | 14. D | 15. A |

Review Questions

1. What is Weka? How to use it?
2. Summarize the tasks done by Weka using a diagram.
3. How to install Weka? Also tell how to use it?
4. Under Weka, we have several tabs for different tasks. Explain each tab.
5. How to pre-process the data in Weka?



Further Readings

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>



Web Links

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Unit 11: Excel Data Analysis

CONTENTS

Objectives

Introduction

11.1 Data Analysis Functions

11.2 Methods for Data Analysis

Summary

Keywords:

Self-Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After studying this unit, you will be able to

- Understand how to analyze data using excel
- Understand some data analysis functions
- Understand the working of data analysis toolpak
- Understand about the descriptive statistics
- Understand how to perform ANOVA

Introduction

Data analysis is the process of cleansing, transforming, and analyzing raw data to obtain usable, relevant information that can assist businesses in making educated decisions. Data analysis is a valuable skill that can help you make better judgments. Microsoft Excel is one of the most used data analysis programs, with the built-in pivot tables being the most popular analytic tool.

11.1 Data Analysis Functions

There are various functions which are used for data analysis. These are shown below:

1. Concatenate()
2. Len()
3. Days()
4. Networkdays()
5. Sumifs()
6. Averageifs()
7. Countsifs()
8. Counta()
9. Vlookup()
10. Hlookup()

11. If()
12. Iferror()
13. Find()/Search()
14. Left()/Right()
15. Rank()

Concatenate()

The concatenate function is used to concatenate the text one after the another.

SYNTAX = CONCATENATE (text1, text2, [text3], ...)

Len()

LEN is used to show the number of characters in each cell.

SYNTAX = LEN (text)

Days()

The number of calendar days between two dates is calculated using this function = DAYS.

SYNTAX =DAYS (end_date, start_date)

Networkdays()

The number of weekends is automatically excluded when using the function. It's classified as a Date/Time Function in Excel.

SYNTAX = NETWORKDAYS (start_date, end_date, [holidays])

Sumifs()

SUM is a familiar formula, but what if you need to sum data based on numerous criteria? It's SUMIFS.

SYNTAX = SUMIFS (sum_range, range1, criteria1, [range2], [criteria2], ...)

Averageifs()

AVERAGEIFS is like SUMIFS.

SYNTAX = AVERAGEIFS (avg_rng, range1, criteria1, [range2], [criteria2], ...)

Countifs()

The COUNTIFS function counts the number of values that satisfy a set of conditions. As a result, it doesn't need a sum range like SUMIFS.

SYNTAX = COUNTIFS (range, criteria)

Counta()

COUNTA determines whether a cell is empty or not. You'll come across incomplete data sets daily as a data analyst. Without needing to restructure the data, COUNTA will allow you to examine any gaps in the dataset.

SYNTAX = COUNTA (value1, [value2], ...)

Vlookup()

The acronym VLOOKUP stands for 'Vertical Lookup.' It's a function that tells Excel to look for a specific value in a column (the so-called 'table array') to return a value from another column in the same row.

SYNTAX = VLOOKUP (lookup_value, table_array, column_index_num, [range_lookup])

Hlookup()

"Horizontal" is represented by the letter H in HLOOKUP. It looks for a value in the top row of a table or an array of values, then returns a value from a row you specify in the table or array in the same column. When your comparison values are in a row across the top of a data table and you wish to look down a specific number of rows, use HLOOKUP. When your comparison values are in a column to the left of the data you wish to find, use VLOOKUP.

SYNTAX = HLOOKUP (lookup_value, table_array, row_index, [range_lookup])

If()

The IF function comes in handy a lot. We can use this function to automate decision-making in our spreadsheets. We could use IF to make Excel conduct a different computation or show a different value based on the results of a logical test (a decision). The IF function will ask you to run a logical test, as well as what action to take if the test is true and what action to take if the test is false.

SYNTAX = IF (logical_test, [value_if_true], [value_if_false])

Iferror()

Two things are required for the IFERROR function to work. What value should be checked for an error and what action should be taken instead.

SYNTAX = IFERROR (value, value_if_error)

Find()/Search()

The FIND function in Excel returns the position of one text string within another (as a number). FIND delivers a #VALUE error if the text cannot be located. However, a =SEARCH for "Bigger" will return results for Bigger or bigger, broadening the scope of the query. This is very helpful when searching for anomalies or unique identifiers. SYNTAX = FIND (find_text, within_text, [start_num]).

SYNTAX = SEARCH (find_text, within_text, [start_num])

Left()/Right()

LEFT and RIGHT are simple and efficient ways for retrieving static data from cells. =RIGHT returns the "x" number of characters from the cell's end, while LEFT returns the "x" number of characters from the cell's beginning.

SYNTAX = LEFT (text, [num_chars])

SYNTAX = RIGHT (text, [num_chars])

Rank()

Even though RANK is an old Excel function, it is nevertheless useful for data analysis. =RANK is a quick way to show how values in a dataset rank in ascending or descending order. RANK is being utilized in this case to determine which clients order the most stuff.

SYNTAX = RANK (number, ref, [order])

11.2 Methods for Data Analysis








- 1) Conditional formatting
- 2) Sorting and filtering
- 3) Pivot tables
- 4) Data visualization in excel

1) Conditional Formatting

Conditional formatting is used to change the appearance of cells in a range based on your specified **conditions**. The conditions are rules based on specified numerical values or matching text. Changing the appearance of cells can visually highlight interesting data points for analysis.

Number of built-in conditions

Conditional Formatting

-  Highlight Cell Rules >
-  Top/Bottom Rules >
-  Data Bars >
-  Colour Scales >
-  Icon Sets >
-  Clear Rules >
-  Manage Rules



Example:

	A	B	C	D
1	Name	Type 1	Speed	
2	Bulbasaur	Grass	45	
3	Ivysaur	Grass	60	
4	Venusaur	Grass	80	
5	Charmander	Fire	65	
6	Charmeleon	Fire	80	
7	Charizard	Fire	100	
8	Squirtle	Water	43	
9	Wartortle	Water	58	
10				

Steps to follow for conditional formatting

- Select the range of values. For example, in the above table shown select the range C2:C9.
- Click on the Conditional Formatting icon in the ribbon, from the **Home** menu.
- Select the **Color Scales** from the drop-down menu. There are 12 Color Scale options with different color variations.
- Click on the "Green - Yellow - Red Colour Scale" icon.

Highlight cell rules

Highlight Cell Rules is a premade type of conditional formatting in Excel used to change the appearance of cells in a range based on your specified **conditions**. The conditions are rules based on specified numerical values, matching text, calendar dates, or duplicated and unique values.

Appearance options

- Light Red Fill with Dark Red Text
- Yellow Fill with Dark Yellow Text
- Green Fill with Dark Green Text
- Light Red Fill
- Red Text
- Red Border

Cell Rule Types

- Greater Than...
- Less Than...
- Between...
- Equal To...
- Text That Contains...
- A Date Occurring...
- Duplicate/Unique Values

2) Sorting and filtering

There are many options for sorting data. You can sort data on common attributes, such as:

- text (A to Z, or Z to A)
- numbers (low to high, or high to low)
- dates and times (newest to oldest, or oldest to newest)
- format (e.g. cell colour).

As well as sorting by individual properties, you can sort data over multiple columns or rows. For example, the following table is initially sorted by ID.

id	title	season	imdb_rating
10	Homer's Night Out	1	7.4
12	Krusty Gets Busted	1	8.3
14	Bart Gets an 'F'	2	8.2
17	Two Cars in Every Garage and Three Eyes on Every Fish	2	8.1
19	Dead Putting Society	2	8.0
35	Blood Feud	2	8.0
37	Mr. Lisa Goes to Washington	3	7.7
39	Bart the Murderer	3	8.7
41	Like Father, Like Clown	3	7.7
44	Saturdays of Thunder	3	7.9

Process of sorting data in Excel:

1. Select a cell in the column you want to sort.
2. In the **Data** tab, go to the **Sort & Filter** group. Then you have two options.
 - To sort values in ascending or descending order based on Excel's interpretation of the column, click the **Sort A to Z** or **Sort Z to A** icons.
 - For more sorting options, click the **Sort** button. You can then specify the **Column**, what to **Sort On**, and **Order**. With the **Add Level** option, you can perform a secondary level of sorting if needed.

Filtering data in excel

You can use filters to temporarily hide some of the data in a table, so you can focus on the data you want to see. When filtering, you can specify exact matches or comparisons ('more than', 'less than') or data that doesn't match specific criteria. The following comparison operators are available in Excel. You can compare two values by using the following operators. When you use these operators to compare two values, the result is a logical value – it's either TRUE or FALSE.

Comparison operator	Meaning	Example	Result (A1=1, B1=2)
=	Equal to	A1=B1	FALSE
>	Greater than	A1>B1	FALSE
<	Less than	A1<B1	TRUE
>=	Greater than or equal to	A1>=B1	FALSE
<=	Less than or equal to	A1<=B1	TRUE
<>	Not equal to	A1<>B1	TRUE

To create a filter in Excel:

1. Select the data you want to work with.
2. Select **Data > Filter** from the ribbon menu.
3. At the top of your selection, select the column header arrow (grey box with downwards arrow).
4. Select **Text Filters** or **Number Filters**, and then select a comparison.
5. Enter the filter criteria and select OK.

3) Pivot Table

Pivot tables are built into Excel. They allow you to group and summarise large quantities of data quickly and easily. If you have an input table with tens, hundreds, or even thousands of rows, pivot tables allow you to extract answers to a series of basic questions about your data with minimal effort. For example: if you have typical sales data, you can use a pivot table to:

- find the sum of total sales per customer
- count the total number of orders by customer
- find the sum total of sales by item type
- create a summary of sales by customer and item type
- find the average amount of sales to a particular customer in a quarter
- create a summary showing the maximum order value by customer and month
- create a breakout summary of orders by customer, month, and item type.

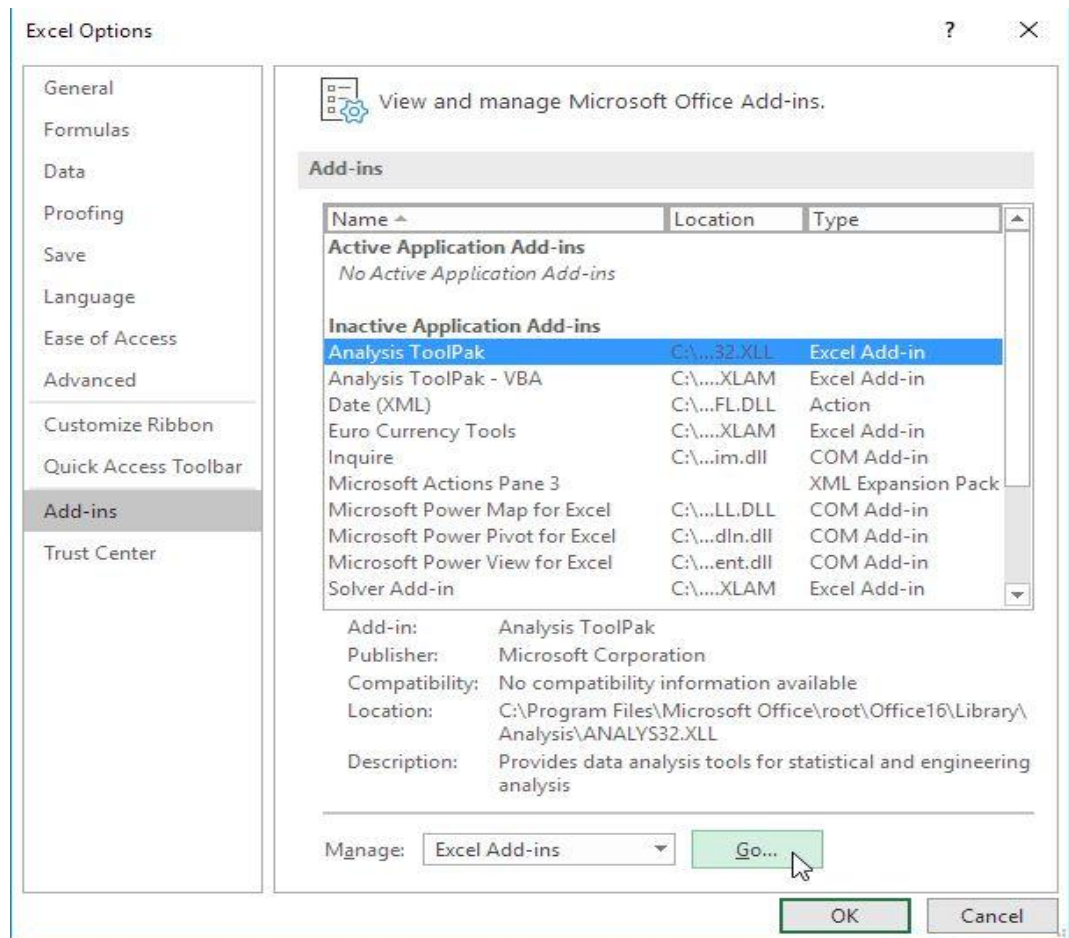
Data Analysis Toolpak

The Analysis ToolPak is an Excel add-in program that provides data analysis tools for financial, statistical and engineering data analysis.

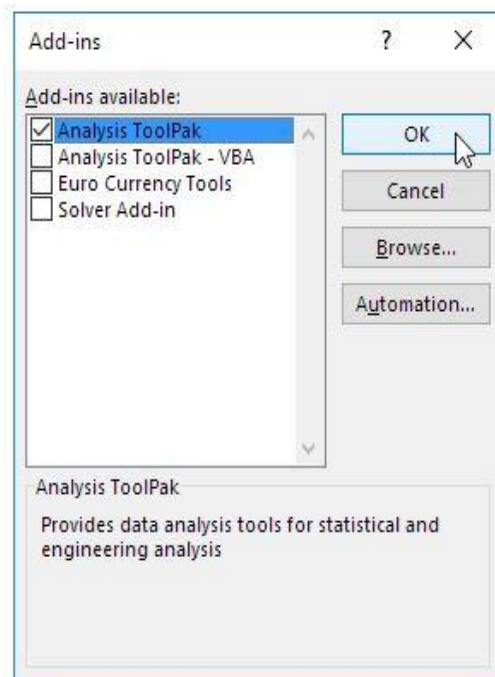
Load the data analysis ToolPak

- 1) On the File tab, click Options.

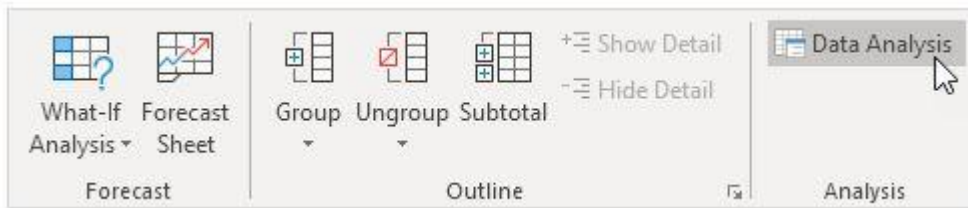
- 2) Under Add-ins, select Analysis ToolPak and click on the Go button.



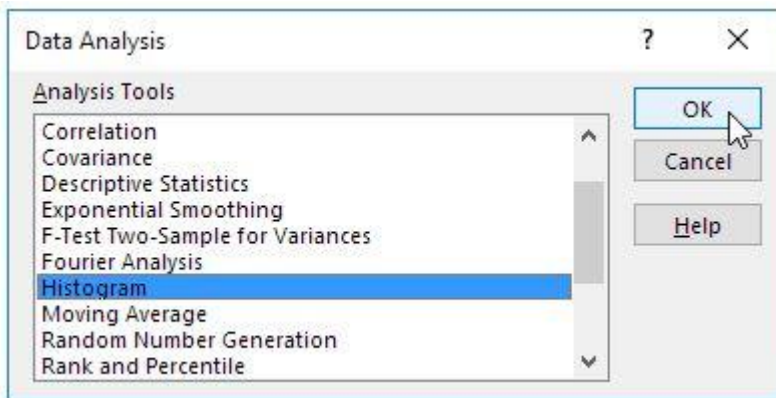
- 3) Check Analysis ToolPak and click on OK



- 4) On the Data tab, in the Analysis group, you can now click on Data Analysis.



5) For example, select Histogram and click OK to create a Histogram in Excel.



Descriptive Statistics

The Descriptive Statistics analysis tool generates a report of univariate statistics for data in the input range, providing information about the central tendency and variability of your data.

ANOVA

The ANOVA analysis tools provide different types of variance analysis. The tool that you should use depends on the number of factors and the number of samples that you have from the populations that you want to test.

Regression

The Regression analysis tool performs linear regression analysis by using the "least squares" method to fit a line through a set of observations. You can analyze how a single dependent variable is affected by the values of one or more independent variables.

Histogram

The Histogram analysis tool calculates individual and cumulative frequencies for a cell range of data and data bins. This tool generates data for the number of occurrences of a value in a data set.

Summary

- Data analysis is a valuable skill that can help you make better judgments. Microsoft Excel is one of the most used data analysis programs, with the built-in pivot tables being the most popular analytic tool.
- The data analysis functions are: Concatenate(), Len(), Days(), Networkdays(), Sumifs(), Averageifs(), Countsifs(), Counta(), Vlookup(), Hlookup(), If(), Iferror(), Find()/Search(), Left()/Right() and Rank().

- “Horizontal” is represented by the letter H in HLOOKUP. It looks for a value in the top row of a table or an array of values, then returns a value from a row you specify in the table or array in the same column
- The syntax of if function is IF (logical_test, [value_if_true], [value_if_false])
- The FIND function in Excel returns the position of one text string within another (as a number). FIND delivers a #VALUE error if the text cannot be located.

Keywords:

- Data analysis: Data analysis is the process of cleansing, transforming, and analyzing raw data to obtain usable, relevant information that can assist businesses in making educated decisions.
- Len(): LEN is used to show the number of characters in each cell.
- COUNTIFS(): The COUNTIFS function counts the number of values that satisfy a set of conditions. As a result, it doesn’t need a sum range like SUMIFS.
- COUNTA(): COUNTA determines whether a cell is empty or not. You’ll come across incomplete data sets daily as a data analyst. Without needing to restructure the data, COUNTA will allow you to examine any gaps in the dataset.
- VLOOKUP(): The acronym VLOOKUP stands for ‘Vertical Lookup.’ It’s a function that tells Excel to look for a specific value in a column (the so-called ‘table array’) to return a value from another column in the same row.

SelfAssessment

1. Data analysis is the process of
 - A. Cleansing
 - B. Transforming
 - C. Analyzing data
 - D. All of the above
2. Which of the following are data analysis functions?
 - A. Len()
 - B. Concatenate()
 - C. Count()
 - D. All of the above
3. Which of the following are data analysis functions?
 - A. Sumif()
 - B. Average ifs()
 - C. Countsifs()
 - D. All of the above
4. What is the syntax of concatenate function for data analysis?
 - A. CONCATENATE(text1, text2, [text3], ...)
 - B. CAT(text1, text2, [text3], ...)
 - C. CONCAT (text1, text2, [text3], ...)
 - D. Any of the above

5. The syntax of days() function is
- A. DAYS (end_date, start_date)
 - B. DAYS (start_date, end_date)
 - C. DAYS (start_date, start_date)
 - D. None of the above
6. What is the syntax of networkdays()?
- A. NETWORKDAYS (end_date, start_date, [holidays])
 - B. NETWORKDAYS (start_date, end_date, [holidays])
 - C. NETWORKDAYS ([holidays], start_date, end_date)
 - D. NETWORKDAYS (start_date, [holidays], end_date)
7. The counta() function allows you to count any gaps in the dataset.
- A. True
 - B. False
8. Which of the following are the methods for data analysis?
- A. Pivot table
 - B. Conditional formatting
 - C. Data visualization
 - D. All of the above
9. Which of the following are the methods for data analysis?
- A. Sorting and filtering
 - B. Pivot table
 - C. Conditional formatting
 - D. All of the above
10. The data analysis toolpak provides the tools for
- A. Financial
 - B. Statistical
 - C. Engineering data analysis
 - D. All of the above
11. In data analysis toolpak, the tools are available for
- A. Correlation
 - B. Histogram
 - C. Rank and percentile
 - D. All of the above
12. In data analysis toolpak, the tools are available for
- A. Exponential smoothing
 - B. Covariance
 - C. Random number generation
 - D. All of the above

13. The descriptive analysis tool provides information about
- Central tendency
 - Variability
 - Both of the above
 - None of the above
14. Which of the following are the data analysis functions?
- If()
 - Vlookup()
 - Hlookup()
 - All of the above
15. Which function is used when the sum is required if a criteria is met?
- SUMCRITERIA()
 - SUM()
 - SUMIF()
 - All of the above

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. D | 2. D | 3. D | 4. A | 5. A |
| 6. B | 7. A | 8. D | 9. D | 10. D |
| 11. D | 12. D | 13. C | 14. D | 15. D |

Review Questions

- What is data analysis? What are the different tools available for this?
- Explain Concatenate(), Len(), Days(), Networkdays() and Sumifs() functions with their syntax.
- Explain averageifs(), countsifs(), counta() and vlookup() functions with their syntax.
- Explain hlookup() and vlookup() functions in detail with example and syntax.
- What are the different methods for data analysis? Explain the use and importance of data analysis toolpak of excel.



Further Readings

<https://www.geeksforgeeks.org/life-cycle-phases-of-data-analytics/>



Web Links

<https://www.simplilearn.com/tutorials/data-science-tutorial/what-is-data-science>

Unit 12: R Tool

CONTENTS

Objectives

Introduction

13.1 Data Types

13.2 Variables

13.3 R operators

13.4 Decision Making

13.5 Loops

13.6 Loop Control Statements

13.7 Functions

13.8 Strings

13.9 R Packages

13.10 Data Reshaping

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this unit, you will be able to

- Understand about R and RStudio
- Understand the R data types
- Understand variables and operators
- Understand the decision-making algorithms and loops
- Understand the functions
- Understand strings and string methods
- Understand R packages

Introduction

R is an open-source programming language mostly used for statistical computing and data analysis. It is available across widely used platforms like Windows, Linux, and MacOS. It generally comes with a command-line interface and provides a vast list of packages for performing tasks. R is an interpreted language that supports both procedural programming and object-oriented programming.

Development of R

- It was designed by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is currently developed by the R Development Core Team.
- R programming language is an implementation of the S programming language.

Why use R programming language?

- R programming is used as a leading tool for machine learning, statistics, and data analysis. Objects, functions, and packages can easily be created by R.
- It's a platform-independent language. This means it can be applied to all operating systems.
- It's an open-source free language. That means anyone can install it in any organization without purchasing a license.
- R programming language is not only a statistic package but also allows us to integrate with other languages (C, C++). Thus, you can easily interact with many data sources and statistical packages.
- The R programming language has a vast community of users and it's growing day by day.
- R is currently one of the most requested programming languages in the Data Science job market that makes it the hottest trend nowadays.

Features of R programming languages

- **Statistical Features of R**
- **Programming Features of R**

Statistical Features of R

- **Basic Statistics:** The most common basic statistical terms are the mean, mode, and median. These are all known as "Measures of Central Tendency." So, using the R language we can measure central tendency very easily.
- **Static graphics:** R is rich with facilities for creating and developing interesting static graphics. R contains functionality for many plot types including graphic maps, mosaic plots, biplots, and the list goes on.
- **Probability distributions:** Probability distributions play a vital role in statistics and by using R we can easily handle various types of probability distribution such as Binomial Distribution, Normal Distribution, Chi-squared Distribution and many more.
- **Data analysis:** It provides a large, coherent, and integrated collection of tools for data analysis

Programming Features of R

- **R Packages:** One of the major features of R is it has a wide availability of libraries. R has CRAN(Comprehensive R Archive Network), which is a repository holding more than 10,000 packages.
- **Distributed Computing:** Distributed computing is a model in which components of a software system are shared among multiple computers to improve efficiency and performance. Two new packages **ddR** and **multidplyr** used for distributed programming in R were released in November 2015.

Advantages of R

- R is the most comprehensive statistical analysis package. As new technology and concepts often appear first in R.
- As R programming language is an open source. Thus, you can run R anywhere and at any time.

- R programming language is suitable for GNU/Linux and Windows operating system.
- R programming is a cross-platform which runs on any operating system.
- In R, everyone is welcome to provide new packages, bug fixes, and code enhancements.

Disadvantages of R

- In the R programming language, the standard of some packages is less than perfect.
- Although, R commands put little pressure on memory management. So, R programming language may consume all available memory.
- In R basically, nobody complains if something doesn't work.
- R programming language is much slower than other programming languages such as Python and MATLAB.

Applications of R

- We use R for Data Science. It gives us a broad variety of libraries related to statistics. It also provides the environment for statistical computing and design.
- R is used by many quantitative analysts as its programming tool. Thus, it helps in data importing and cleaning.
- R is the most prevalent language. So many data analysts and research programmers use it. Hence, it is used as a fundamental tool for finance.
- Tech giants like Google, Facebook, Bing, Twitter, Accenture, Wipro and many more use R nowadays.

Interesting facts about R

- R programming language is an implementation of the S programming language. It also combines with lexical scoping semantics inspired by Scheme. It is named partly after the first names of the first two R authors and partly as a play on the name of S.
- R supports both procedural programming and object-oriented programming. Procedural programming includes the procedure, records, modules, and procedure calls. While object-oriented programming language includes class, objects, and generic functions
- R language is an interpreted language instead of a compiled language. Therefore, it doesn't need a compiler to compile code into an executable program. This makes running an R script much less time-consuming.
- The number of R packages available either through CRAN or GitHub is 1,00,000 and they do epic stuff with just one line of code. It could range from Regression to Bayesian analysis.
- R is growing faster than any other data science language. It's the most-used data science language after SQL. It is used by 70% of data miners.
- One of the packages in R namely rmarkdown package helps you create reproducible Word documents and reproducible Powerpoint Presentations from your R markdown code just by changing one line in the YAML! ("YAML Ain't Markup Language!")
- It is really very easy in R to connect to almost any database using the dbplyr package. This makes it possible for an R user to work independently and pulling data from almost all common database types. You can also use packages like bigrquery to work directly with BigQuery and other high-performance data stores.
- You can build and host interactive web apps in just a few lines of code in R. Using the flexdashboard package in R you can create interactive web apps with a few lines of code. And using the rconnect package you can also host your web apps on your own server or, even easier, host them on a cloud server.

- You can not only deploy web apps but also can make them into awesome video games in R. The nesy package helps you create NES(The Nintendo Entertainment System) looking Shiny apps and deploy them just like you would any other Shiny app.
- You can build APIs and serve them from R. The plumber package in R helps you convert R functions to web APIs that can be integrated into downstream applications.

Environment in R

- The environment is a virtual space that is triggered when an interpreter of a programming language is launched.
- Simply, the environment is a collection of all the objects, variables, and functions. Or Environment can be assumed as a top-level object that contains the set of names/variables associated with some values.

Introduction to R studio

R Studio is an integrated development environment(IDE) for R. IDE is a GUI, where you can write your codes, see the results, and see the variables that are generated during the course of programming. R Studio is available as both Open source and Commercial software. R Studio is also available as both Desktop and Server versions. R Studio is also available for various platforms such as Windows, Linux, and macOS. R Studio is an open-source tool that provides IDE to use R language, and enterprise-ready professional software for data science teams to develop share the work with their team. R Studio can be downloaded from its official website (<https://rstudio.com/>)

13.1 Data Types

Variables are nothing but reserved memory locations to store values. This means that, when you create a variable, you reserve some space in memory. You may like to store information of various data types like character, wide character, integer, floating point, double floating point, Boolean etc. Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory. In contrast to other programming languages like C and java in R, the variables are not declared as some data type. The variables are assigned with R-Objects and the data type of the R-object becomes the data type of the variable. There are many types of R-objects.

Various types of Data Types

- Vectors
- Lists
- Matrices
- Arrays
- Factors
- Data Frames

Vector object

The simplest of these objects is the **vector object** and there are six data types of these atomic vectors, also termed as six classes of vectors.

- 1) Logical – True, False
- 2) Numeric – 12.3, 5, 999
- 3) Integer – 2L, 3L, 0L
- 4) Complex – 3+2i
- 5) Character – 'a', "good", "TRUE", '23.4'

6) Raw - "Hello" is stored as 48 65 6c 6c 66

When you want to create vector with more than one element, you should use `c()` function which means to combine the elements into a vector.

- # Create a vector

```
apple <- c('red', 'apple', 'yellow')
print(apple)
```

- # Get the class of a vector

```
print(class(apple))
```

Lists

A list is an R-object which can contain many different types of elements inside it like vectors, functions and even another list inside it.

- #Create a list

```
list1 <- list(c(2,5,3),21.3,sin)
```

- #Print the list

```
Print(list1)
```

Matrix

A matrix is a two-dimensional rectangular data set. It can be created using a vector input to the `matrix` function.

- #Create a matrix

```
M = matrix(c('a', 'a', 'b', 'c', 'b', 'a'), nrow=2, ncol=3, byrow=TRUE)
```

```
Print(M)
```

Arrays

While matrices are confined to two dimensions, arrays can be of any number of dimensions. The `array` function takes a `dim` attribute which creates the required number of dimensions.

- #Create an array

```
a <- array(c('green', 'yellow'), dim=c(3,3,2))
```

```
print(a)
```

Factors

Factors are the R-objects which are created using a vector. It stores the vector along with the distinct values of the elements in the vector as labels. The labels are always character irrespective of whether it is numeric or character or Boolean etc. in the input vector. They are useful in statistical modeling. Factors are created using the `factor()` function. The `nlevels` function gives the count of levels.

- #Create a vector

```
apple_colors <- c('green', 'green', 'yellow', 'red', 'red', 'red', 'green')
```

- #Create a factor object

```
factor_apple <- factor(apple_colors)
```

- Print the factor

```
print(factor_apple)
print(nlevels(factor_apple))
```

Data Frames

Data frames are tabular data objects. Unlike a matrix in data frame each column can contain different modes of data. The first column can be numeric while the second column can be character and third column can be logical. It is a list of vectors of equal length. Data Frames are created using the **data.frame()** function.

- #Create the data frame

```
BMI <- data.frame(gender = c("Male", "Male", "Female"),
                 height = c(152, 171.5, 165),
                 weight = c(81, 93, 78),
                 age = c(42, 38, 26) )

print(BMI)
```

13.2 Variables

A variable provides us with named storage that our programs can manipulate. A variable in R can store an atomic vector, group of atomic vectors or a combination of many R objects. A valid variable name consists of letters, numbers and the dot or underline characters. The variable name starts with a letter, or the dot not followed by a number.

Valid variable names

- var_name2.
- .var_name,
- var.name

Invalid variable names

- var_name%
- 2var_name
- .2var_name
- _var_name

Variable assignment

The variables can be assigned values using leftward, rightward, and equal to operator. The values of the variables can be printed using **print()** or **cat()** function. The **cat()** function combines multiple items into a continuous print output.

Assignment with equal operator

- var.1 = c(0,1,2,3)

Assignment with leftward operator

- var.2 <- c("learn", "R")

Assignment with rightward operator

- C(TRUE,1) -> var.3

Printing of variables

```
cat("var.1 is ", var.1, "\n")
cat("var.2 is ", var.2, "\n")
cat("var.3 is ", var.3, "\n")
```

Data type of a variable

In R, a variable itself is not declared of any data type, rather it gets the data type of the R - object assigned to it. So, R is called a dynamically typed language, which means that we can change a variable's data type of the same variable again and again when using it in a program.

```
var_x<- "Hello"
cat("The class of var_x is ",class(var_x),"\n")
var_x<- 34.5
cat(" Now the class of var_x is ",class(var_x),"\n")
var_x<- 27L
cat(" Next the class of var_x becomes ",class(var_x),"\n")
```

Finding variables

To know all the variables currently available in the workspace we use the **ls()** function. Also, the **ls()** function can use patterns to match the variable names.

- Print(ls())

Deleting variables

Variables can be deleted by using the **rm()** function. Below we delete the variable var.3. On printing the value of the variable error is thrown.

```
rm(var.3)
print(var.3)
```

13.3 R operators

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. R language is rich in built-in operators and provides the following types of operators.

Types of operators

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Assignment Operators
- Miscellaneous Operators

Arithmetic operators

- + Addition

- - Subtraction
- * Multiplication
- / Division
- %% Remainder
- ^ Power

Relational Operators

- > Greater than
- < Less than
- == Equal to
- <= Less than or equal to
- >= Greater than or equal to
- != Not equal to

Logical Operators

- & Element wise Logical AND
- | Element wise Logical OR
- ! Element wise Logical NOT
- && Logical AND
- || Logical OR

Assignment operators

- <- or == or <<- Left assignment
- -> or ->> Right assignment

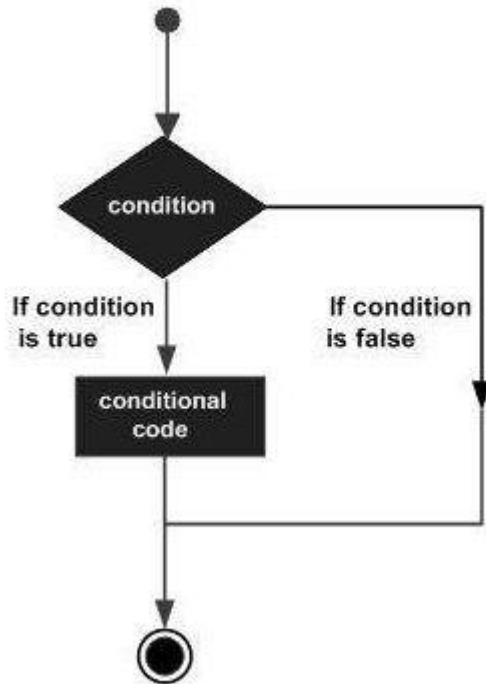
Miscellaneous operator

- : Colon operator
- %in% Identify an element belongs to a vector
- %*% Multiply a matrix with its transpose

13.4 Decision Making

Decision making structures require the programmer to specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be **true**, and optionally, other statements to be executed if the condition is determined to be **false**.

General form



Types of decision-making statements

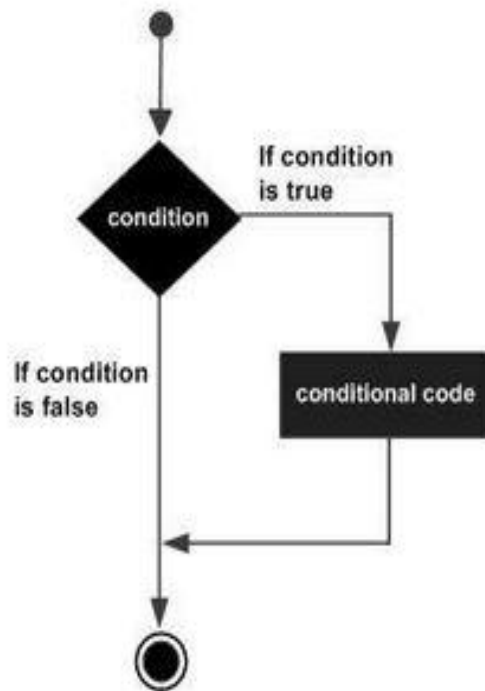
- If statement
- If.....else statement
- Switch statement

If statement

An **if** statement consists of a Boolean expression followed by one or more statements. The syntax is:

```
if(boolean_expression) {  
    // statement(s) will execute if the boolean expression is true.  
}
```

If the Boolean expression is evaluated to be **true**, then the block of code inside the if statement will be executed. If Boolean expression evaluates to be **false**, then the first set of code after the end of the if statement (after the closing curly brace) will be executed.



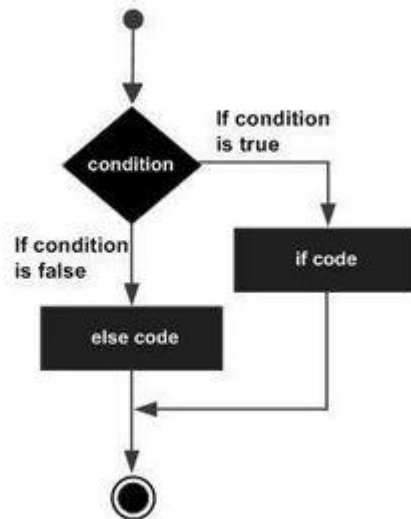
```
x <- 30L
if(is.integer(x)) {
  print("X is an Integer")
}
Output: [1] "X is an Integer"
```

If...Else statement

An **if** statement can be followed by an optional **else** statement which executes when the boolean expression is false. The syntax is:

```
if(boolean_expression) {
  // statement(s) will execute if the boolean expression is true.
} else {
  // statement(s) will execute if the boolean expression is false.
}
```

If the Boolean expression evaluates to be **true**, then the **if block** of code will be executed, otherwise **else block** of code will be executed.



```

x <- c("what","is","truth")
if("Truth" %in% x) {
  print("Truth is found")
} else {
  print("Truth is not found")
}
O/P: [1] "Truth is not found"

```

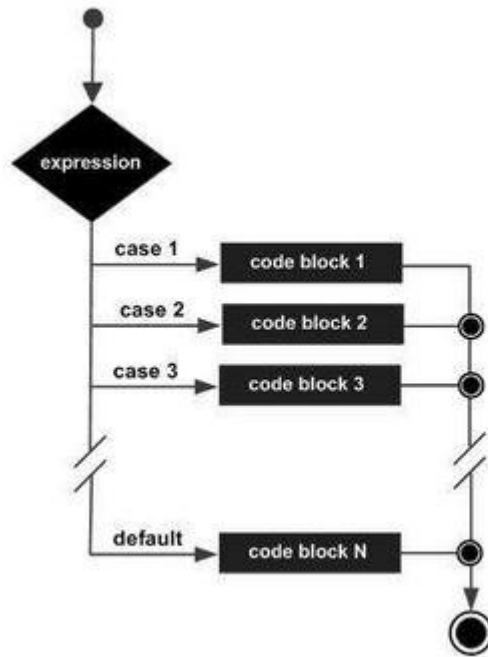
Switch statement

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case. The syntax is:

```
switch(expression, case1, case2, case3....)
```

Rules apply to a switch statement –

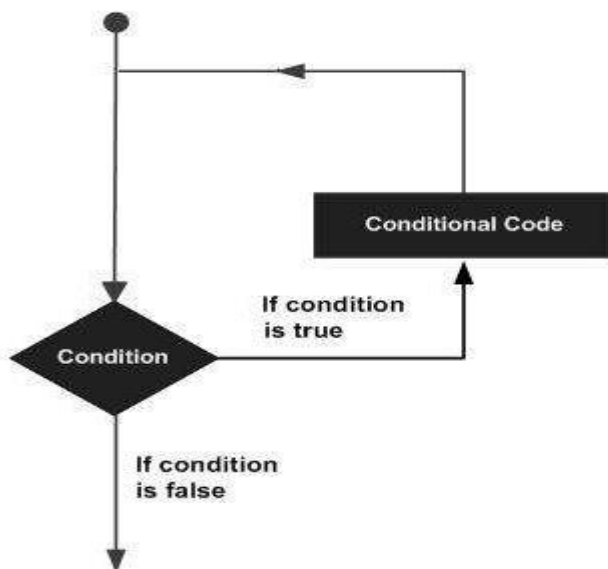
- If the value of expression is not a character string it is coerced to integer.
- You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.
- If the value of the integer is between 1 and `nargs()-1` (The max number of arguments) then the corresponding element of case condition is evaluated, and the result returned.
- If expression evaluates to a character string, then that string is matched (exactly) to the names of the elements.
- If there is more than one match, the first matching element is returned.
- No Default argument is available.
- In the case of no match, if there is an unnamed element of ... its value is returned. (If there is more than one such argument an error is returned.)



```
x <- switch(3, "first", "second", "third", "fourth")
print(x)
O/P: [1] "third"
```

13.5 Loops

There may be a situation when you need to execute a block of code several number of times. In general, statements are executed sequentially. The first statement in a function is executed first, followed by the second, and so on. Programming languages provide various control structures that allow for more complicated execution paths. A loop statement allows us to execute a statement or group of statements multiple times and the following is the general form of a loop statement in most of the programming languages.



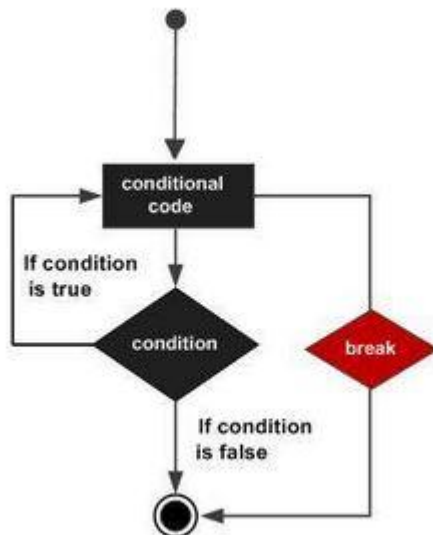
Kinds of Loops

- Repeat loop
- While loop
- For loop

Repeat Loop

The **Repeat loop** executes the same code again and again until a stop condition is met. The syntax is:

```
repeat {
  commands
  if(condition) {
    break
  }
}
```



```
v <- c("Hello","loop")
cnt<- 2
repeat {
  print(v)
  cnt<- cnt+1
  if(cnt> 5) {
    break
  }
}
```

O/P: [1] "Hello" "loop"

[1] "Hello" "loop"

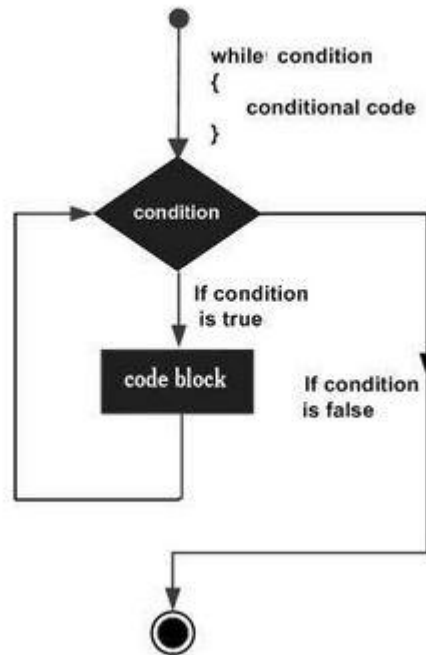
```
[1] "Hello" "loop"
```

```
[1] "Hello" "loop"
```

While Loop

The While loop executes the same code again and again until a stop condition is met. The syntax is:

```
while (test_expression) {
  statement
}
```



Here key point of the **while** loop is that the loop might not ever run. When the condition is tested and the result is false, the loop body will be skipped and the first statement after the while loop will be executed.

```
v <- c("Hello","while loop")
cnt<- 2
while (cnt< 7) {
  print(v)
  cnt = cnt + 1
}
```

```
[1] "Hello" "while loop"
```

```
[1] "Hello" "while loop"
```

```
[1] "Hello" "while loop"
```

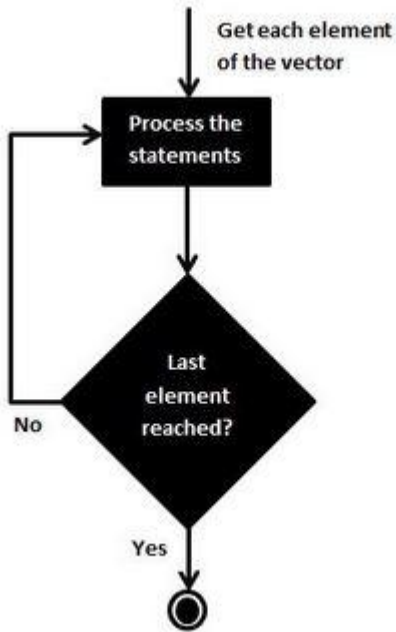
```
[1] "Hello" "while loop"
```

```
[1] "Hello" "while loop"
```

For loop

A **For loop** is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

```
for (value in vector) {
  statements
}
```



R's for loops are particularly flexible in that they are not limited to integers, or even numbers in the input. We can pass character vectors, logical vectors, lists or expressions.

```
v <- LETTERS[1:4]
for (i in v) {
  print(i)
}
```

O/P: [1] "A"

[1] "B"

[1] "C"

[1] "D"

13.6 Loop Control Statements

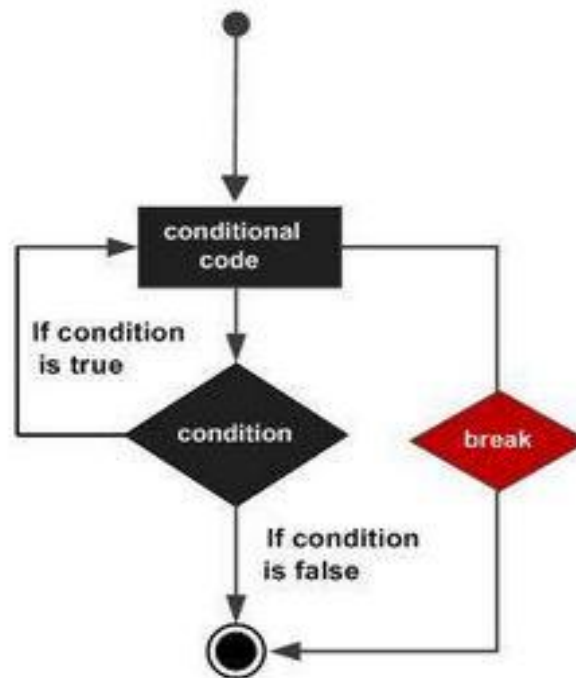
Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed. The loop control statements are:

- Break statement
- Next statement

Break Statement

The break statement in R programming language has the following two usages.

- 1) When the break statement is encountered inside a loop, the loop is immediately terminated, and program control resumes at the next statement following the loop.
- 2) It can be used to terminate a case in the switch statement.



```

v <- c("Hello","loop")
cnt<- 2
repeat {
  print(v)
  cnt<- cnt + 1
  if(cnt> 5) {
    break
  }
}
  
```

O/P: [1] "Hello" "loop"

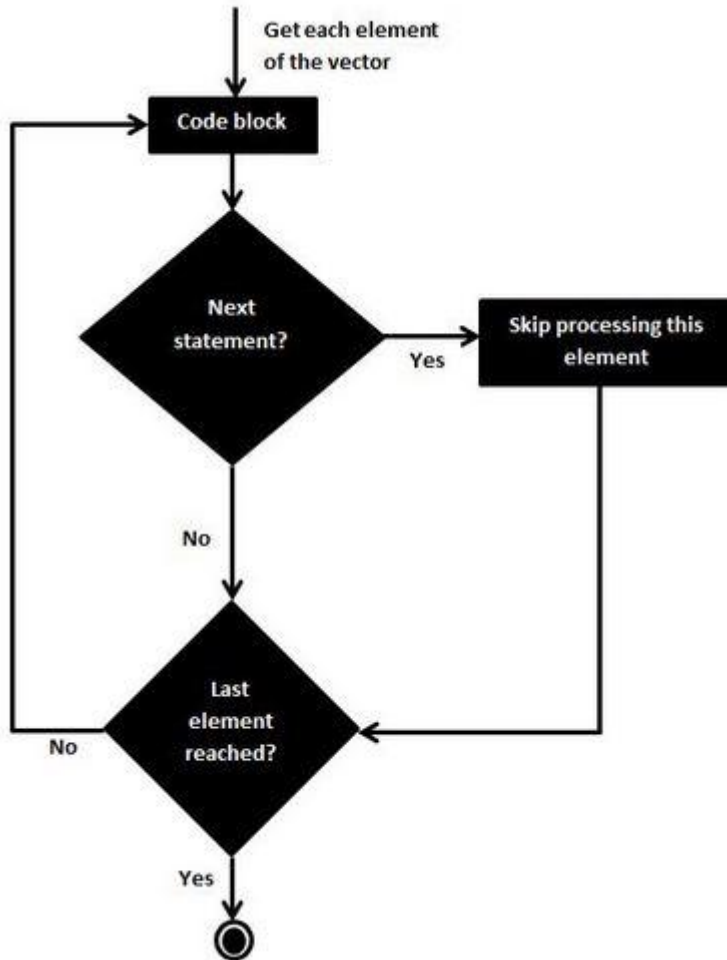
[1] "Hello" "loop"

[1] "Hello" "loop"

[1] "Hello" "loop"

Next Statement

The **next** statement in R programming language is useful when we want to skip the current iteration of a loop without terminating it. On encountering next, the R parser skips further evaluation and starts the next iteration of the loop.



```

v <- LETTERS[1:6]
for (i in v) {
  if (i == "D") {
    next
  }
  print(i)
}
  
```

O/P: [1] "A"

[1] "B"

[1] "C"

[1] "E"

[1] "F"

13.7 Functions

A function is a set of statements organized together to perform a specific task. R has many in-built functions, and the user can create their own functions. In R, a function is an object so the R interpreter can pass control to the function, along with arguments that may be necessary for the function to accomplish the actions. The function in turn performs its task and returns control to the interpreter as well as any result which may be stored in other objects.

Function Definition

An R function is created by using the keyword **function**. The basic syntax of an R function definition is:

```
function_name <- function(arg_1, arg_2, ...) {  
  function body  
}
```

Function Components

- **Function Name** – This is the actual name of the function. It is stored in an R environment as an object with this name.
- **Arguments** – An argument is a placeholder. When a function is invoked, you pass a value to the argument. Arguments are optional; that is, a function may contain no arguments. Also, arguments can have default values.
- **Function Body** – The function body contains a collection of statements that defines what the function does.
- **Return Value** – The return value of a function is the last expression in the function body to be evaluated

R has many **in-built** functions which can be directly called in the program without defining them first. We can also create and use our own functions referred to as **user defined** functions.

Built-in Functions

The simple examples of in-built functions are `seq()`, `mean()`, `max()`, `sum(x)` and `paste(...)` etc. They are directly called by user written programs.

- # Create a sequence of numbers from 32 to 44.

```
print(seq(32,44))
```

- # Find mean of numbers from 25 to 82.

```
print(mean(25:82))
```

- # Find sum of numbers from 41 to 68.

```
print(sum(41:68))
```

O/P: [1] 32 33 34 35 36 37 38 39 40 41 42 43 44

[1] 53.5

[1] 1526

User Defined Functions

We can create user-defined functions in R. They are specific to what a user wants and once created they can be used like the built-in functions.

- # Create a function to print squares of numbers in sequence.

```
new.function<- function(a) {
  for(i in 1:a) {
    b <- i^2
    print(b)
  }
}
```

- Calling a function

Call the function new.function supplying 6 as an argument.

```
new.function(6)
```

O/P: [1] 1

[1] 4

[1] 9

[1] 16

[1] 25

[1] 36

- Calling a function without an argument

Create a function without an argument.

```
new.function<- function() {
  for(i in 1:5) {
    print(i^2)
  }
}
```

- # Call the function without supplying an argument.

```
new.function()
```

O/P: [1] 1
[1] 4
[1] 9
[1] 16
[1] 25

The arguments to a function call can be supplied in the same sequence as defined in the function or they can be supplied in a different sequence but assigned to the names of the arguments.

- # Create a function with arguments.

```
new.function<- function(a,b,c) {  
  result <- a * b + c  
  print(result)  
}
```

- # Call the function by position of arguments.

```
new.function(5,3,11)
```

- # Call the function by names of the arguments.

```
new.function(a = 11, b = 5, c = 3)
```

- Calling a function with default arguments

We can define the value of the arguments in the function definition and call the function without supplying any argument to get the default result. But we can also call such functions by supplying new values of the argument and get non default result.

- # Create a function with arguments.

```
new.function<- function(a = 3, b = 6) {  
  result <- a * b  
  print(result)  
}
```

- # Call the function without giving any argument.

```
new.function()
```

- # Call the function giving new values of the argument.

```
new.function(9,5)
```

O/P: [1] 18

[1] 45

13.8 Strings

Any value written within a pair of single quotes or double quotes in R is treated as a string. Internally R stores every string within double quotes, even when you create them with single quote.

Rules applied in string construction

- The quotes at the beginning and end of a string should be both double quotes or both single quote. They cannot be mixed.
- Double quotes can be inserted into a string starting and ending with single quote.
- Single quotes can be inserted into a string starting and ending with double quotes.
- Double quotes cannot be inserted into a string starting and ending with double quotes.
- Single quotes cannot be inserted into a string starting and ending with single quote.

Valid strings

```
a <- 'Start and end with single quote'
print(a)
b <- "Start and end with double quotes"
print(b)
c <- "single quote ' in between double quotes"
print(c)
d <- 'Double quotes " in between single quote'
print(d)
O/P: [1] "Start and end with single quote"
[1] "Start and end with double quotes"
[1] "single quote ' in between double quote"
[1] "Double quote \" in between single quote"
```

Invalid strings

```
e <- 'Mixed quotes"
print(e)
f <- 'Single quote ' inside single quote'
print(f)
g <- "Double quotes " inside double quotes"
```

```
print(g)
Error: unexpected symbol in:
"print(e)
f <- 'Single"
Execution halted
```

String manipulation

- Concatenating Strings - paste () function
- Formatting numbers & strings - format() function
- Counting number of characters in a string - nchar() function
- Changing the case - toupper() & tolower() functions
- Extracting parts of a string - substring () function

Concatenating Strings - paste() function

Many strings in R are combined using the **paste()** function. It can take any number of arguments to be combined.

```
paste (... , sep = " ", collapse = NULL)
```

- ... represents any number of arguments to be combined.
- **sep** represents any separator between the arguments. It is optional.
- **collapse** is used to eliminate the space in between two strings. But not the space within two words of one string.

```
a <- "Hello"
b <- 'How'
c <- "are you? "
print(paste(a,b,c))
print(paste(a,b,c, sep = "-"))
print(paste(a,b,c, sep = "", collapse = ""))
```

```
O/P: [1] "Hello How are you? "
```

```
[1] "Hello-How-are you? "
```

```
[1] "HelloHoware you? "
```

Formatting numbers & strings - format() function

Numbers and strings can be formatted to a specific style using **format()** function.

```
format(x, digits, nsmall, scientific, width, justify = c("left", "right", "centre", "none"))
```

- **x** is the vector input.

- **digits** are the total number of digits displayed.
- **nsmall** is the minimum number of digits to the right of the decimal point.
- **scientific** is set to TRUE to display scientific notation.
- **width** indicates the minimum width to be displayed by padding blanks in the beginning.
- **Justify** is the display of the string to left, right or center.

Total number of digits displayed. Last digit rounded off.

```
result <- format(23.123456789, digits = 9)
```

```
print(result)
```

- # Display numbers in scientific notation.

```
result <- format(c(6, 13.14521), scientific = TRUE)
```

```
print(result)
```

- # The minimum number of digits to the right of the decimal point.

```
result <- format(23.47, nsmall = 5)
```

```
print(result)
```

- # Format treats everything as a string.

```
result <- format(6)
```

```
print(result)
```

- # Numbers are padded with blank in the beginning for width.

```
result <- format(13.7, width = 6)
```

```
print(result)
```

- # Left justify strings.

```
result <- format("Hello", width = 8, justify = "l")
```

```
print(result)
```

- # Justify string with center.

```
result <- format("Hello", width = 8, justify = "c")
```

```
print(result)
```

```
O/P: [1] "23.1234568"
```

```
[1] "6.000000e+00" "1.314521e+01"
```

```
[1] "23.47000"
```

```
[1] "6"
```

```
[1] " 13.7"
```

```
[1] "Hello "
```

```
[1] " Hello "
```


Counting number of characters in a string - nchar() function

This function counts the number of characters including spaces in a string. The syntax is:

```
nchar(x)
```

x is the vector input.

```
result <- nchar("Count the number of characters")
```

```
print(result)
```

```
O/P: [1] 30
```

Changing the case - toupper() & tolower() functions

These functions change the case of characters of a string.

```
toupper(x)
```

```
tolower(x)
```

- x is the vector input.
- # Changing to Upper case.

```
result <- toupper("Changing to Upper")
```

```
print(result)
```

- # Changing to lower case.

```
result <- tolower("Changing to Lower")
```

```
print(result)
```

- O/P: [1] "CHANGING TO UPPER"

```
[1] "changing to lower"
```

Extracting parts of a string - substring() function

This function extracts parts of a String.

- substring(x,first,last)

Following is the description of the parameters used –

- x is the character vector input.
- first is the position of the first character to be extracted.
- last is the position of the last character to be extracted
- # Extract characters from 5th to 7th position.

```
result <- substring("Extract", 5, 7)
```

```
print(result)
```

```
O/P: [1] "act"
```

13.9 R Packages

R packages are a collection of R functions, compiled code and sample data. They are stored under a directory called "**library**" in the R environment. By default, R installs a set of packages during installation. More packages are added later when they are needed for some specific purpose. When we start the R console, only the default packages are available by default. Other packages which are already installed must be loaded explicitly to be used by the R program that is going to use them.

Check available R packages

- Get library locations containing R packages: `.libPaths()`
- Get the list of all the packages installed: `library()`

Install a new package

There are two ways to add new R packages. One is installing directly from the CRAN directory, and another is downloading the package to your local system and installing it manually.

Install from CRAN

```
install.packages("Package Name")
```

```
# Install the package named "XML".
```

```
install.packages("XML")
```

Install package manually

```
install.packages(file_name_with_path, repos = NULL, type = "source")
```

```
# Install the package named "XML"
```

```
install.packages("E:/XML_3.98-1.3.zip", repos = NULL, type = "source")
```

Load package to library

Before a package can be used in the code, it must be loaded to the current R environment. You also need to load a package that is already installed previously but not available in the current environment. A package is loaded using the following command –

```
library ("package Name", lib.loc = "path to library")
```

```
# Load the package named "XML"
```

```
install.packages("E:/XML_3.98-1.3.zip", repos = NULL, type = "source")
```

13.10 Data Reshaping

Data Reshaping in R is about changing the way data is organized into rows and columns. Most of the time data processing in R is done by taking the input data as a data frame. Data Reshaping in R is about changing the way data is organized into rows and columns. Most of the time data processing in R is done by taking the input data as a data frame.

Joining Columns and Rows in a Data Frame

We can join multiple vectors to create a data frame using the `cbind()` function. Also, we can merge two data frames using `rbind()` function.

Merging data frames

We can merge two data frames by using the `merge()` function. The data frames must have the same column names on which the merging happens.

Melting and casting

One of the most interesting aspects of R programming is about changing the shape of the data in multiple steps to get a desired shape. The functions used to do this are called `melt()` and `cast()`.

Summary

- R is an open-source programming language mostly used for statistical computing and data analysis. It is available across widely used platforms like Windows, Linux, and MacOS.
- R programming is used as a leading tool for machine learning, statistics, and data analysis. Objects, functions, and packages can easily be created by R.
- As R programming language is an open source. Thus, you can run R anywhere and at any time.
- Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory.
- The data types in R are: vectors, lists, matrices, arrays, factors and data frames.
- Factors are the R-objects which are created using a vector. It stores the vector along with the distinct values of the elements in the vector as labels.
- A valid variable name consists of letters, numbers and the dot or underline characters.
- The variables can be assigned values using leftward, rightward and equal to operator.
- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. R language is rich in built-in operators and provides the following types of operators.
- A function is a set of statements organized together to perform a specific task. R has many in-built functions, and the user can create their own functions.

Keywords

- **R:** R is an interpreted language that supports both procedural programming and object-oriented programming. This is an implementation of the S programming language.
- **RStudio:** R Studio is an integrated development environment (IDE) for R. IDE is a GUI, where you can write your quotes, see the results, and also see the variables that are generated during the course of programming.

- **R Objects:** The variables are assigned with R-Objects and the data type of the R-object becomes the data type of the variable. There are many types of R-objects.
- **Lists:** A list is an R-object which can contain many different types of elements inside it like vectors, functions and even another list inside it.
- **Variable in R:** A variable provides us with named storage that our programs can manipulate. A variable in R can store an atomic vector, group of atomic vectors or a combination of many Robjects.
- **Loops:** A loop statement allows us to execute a statement or group of statements multiple times and the following is the general form of a loop statement in most of the programming languages.

SelfAssessment

1. R language is used for
 - A. Statistical computing
 - B. Data analysis
 - C. Fixing the bugs
 - D. All of the above

2. R language is available across
 - A. Windows
 - B. MAC
 - C. Linux
 - D. All of the above

3. R is an interpreted language which supports
 - A. Only procedural programming
 - B. Object-oriented programming
 - C. Both procedural and object-oriented programming
 - D. None of the above

4. Which of the following are the data types in R programming language?
 - A. Vectors
 - B. Lists
 - C. Data frames
 - D. All of the above mentioned

5. Which attribute creates the required number of dimensions in arrays?
 - A. dimension
 - B. dim
 - C. sides
 - D. All of the above

6. The values of the variables can be printed using
 - A. Cat()
 - B. Print()
 - C. Both above mentioned
 - D. None of the above

7. The decision-making structures in R programming languages are
 - A. If statement
 - B. If-else statement
 - C. Switch statement
 - D. All the above

8. In switch statement, the number of cases allowed are:
 - A. 1
 - B. 2
 - C. 3
 - D. Any

9. Each case in switch statement is followed by the value to be compared to and a
 - A. Colon
 - B. Comma
 - C. Semi-colon
 - D. Asterisk

10. The kinds of loops in R language are
 - A. While loop
 - B. Repeat loop
 - C. For loop
 - D. All of the above

11. Loop control statements in R are
 - A. Break statement
 - B. Next statement
 - C. Both of the above
 - D. None of the above

12. Which of the following are the function components?
 - A. Function body
 - B. Function name
 - C. Arguments
 - D. All of the above mentioned

13. The strings in R programming language can be written in
 - A. Single quotes
 - B. Double quotes
 - C. Either single or double quotes
 - D. None of the above

14. The concatenation of strings can be done using
 - A. paste()
 - B. format()
 - C. nchar()
 - D. substring()

15. The numbers and strings can be formatted to a specific style using
 - A. paste()
 - B. format()
 - C. nchar()
 - D. substring()

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. D | 2. D | 3. C | 4. D | 5. B |
| 6. D | 7. D | 8. D | 9. A | 10. D |
| 11. C | 12. D | 13. D | 14. A | 15. B |

Review Questions

1. Why is R programming language used? Also explain the features of R programming language.
2. What are the advantages and disadvantages of R programming language?
3. What is a data type? Which data types exist in R programming language?
4. What is a vector object? How do we create a vector, and get the class of a vector?
5. What are operators? Explain its types.
6. What is decision making structures in R programming language? Explain.

**Further Readings**

<https://www.r-project.org/about.html>

<https://www.geeksforgeeks.org/r-programming-language-introduction/>

Unit 13: R Tool

CONTENTS

Objectives

Introduction

13.1 Data Types

13.2 Variables

13.3 R operators

13.4 Decision Making

13.5 Loops

13.6 Loop Control Statements

13.7 Functions

13.8 Strings

13.9 R Packages

13.10 Data Reshaping

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After this unit, you will be able to

- Understand about R and RStudio
- Understand the R data types
- Understand variables and operators
- Understand the decision-making algorithms and loops
- Understand the functions
- Understand strings and string methods
- Understand R packages

Introduction

R is an open-source programming language mostly used for statistical computing and data analysis. It is available across widely used platforms like Windows, Linux, and MacOS. It generally comes with a command-line interface and provides a vast list of packages for performing tasks. R is an interpreted language that supports both procedural programming and object-oriented programming.

Development of R

- It was designed by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand, and is currently developed by the R Development Core Team.
- R programming language is an implementation of the S programming language.

Why use R programming language?

- R programming is used as a leading tool for machine learning, statistics, and data analysis. Objects, functions, and packages can easily be created by R.
- It's a platform-independent language. This means it can be applied to all operating systems.
- It's an open-source free language. That means anyone can install it in any organization without purchasing a license.
- R programming language is not only a statistic package but also allows us to integrate with other languages (C, C++). Thus, you can easily interact with many data sources and statistical packages.
- The R programming language has a vast community of users and it's growing day by day.
- R is currently one of the most requested programming languages in the Data Science job market that makes it the hottest trend nowadays.

Features of R programming languages

- **Statistical Features of R**
- **Programming Features of R**

Statistical Features of R

- **Basic Statistics:** The most common basic statistical terms are the mean, mode, and median. These are all known as "Measures of Central Tendency." So, using the R language we can measure central tendency very easily.
- **Static graphics:** R is rich with facilities for creating and developing interesting static graphics. R contains functionality for many plot types including graphic maps, mosaic plots, biplots, and the list goes on.
- **Probability distributions:** Probability distributions play a vital role in statistics and by using R we can easily handle various types of probability distribution such as Binomial Distribution, Normal Distribution, Chi-squared Distribution and many more.
- **Data analysis:** It provides a large, coherent, and integrated collection of tools for data analysis

Programming Features of R

- **R Packages:** One of the major features of R is it has a wide availability of libraries. R has CRAN(Comprehensive R Archive Network), which is a repository holding more than 10,000 packages.
- **Distributed Computing:** Distributed computing is a model in which components of a software system are shared among multiple computers to improve efficiency and performance. Two new packages **ddR** and **multidplyr** used for distributed programming in R were released in November 2015.

Advantages of R

- R is the most comprehensive statistical analysis package. As new technology and concepts often appear first in R.
- As R programming language is an open source. Thus, you can run R anywhere and at any time.

- R programming language is suitable for GNU/Linux and Windows operating system.
- R programming is a cross-platform which runs on any operating system.
- In R, everyone is welcome to provide new packages, bug fixes, and code enhancements.

Disadvantages of R

- In the R programming language, the standard of some packages is less than perfect.
- Although, R commands put little pressure on memory management. So, R programming language may consume all available memory.
- In R basically, nobody complains if something doesn't work.
- R programming language is much slower than other programming languages such as Python and MATLAB.

Applications of R

- We use R for Data Science. It gives us a broad variety of libraries related to statistics. It also provides the environment for statistical computing and design.
- R is used by many quantitative analysts as its programming tool. Thus, it helps in data importing and cleaning.
- R is the most prevalent language. So many data analysts and research programmers use it. Hence, it is used as a fundamental tool for finance.
- Tech giants like Google, Facebook, Bing, Twitter, Accenture, Wipro and many more use R nowadays.

Interesting facts about R

- R programming language is an implementation of the S programming language. It also combines with lexical scoping semantics inspired by Scheme. It is named partly after the first names of the first two R authors and partly as a play on the name of S.
- R supports both procedural programming and object-oriented programming. Procedural programming includes the procedure, records, modules, and procedure calls. While object-oriented programming language includes class, objects, and generic functions
- R language is an interpreted language instead of a compiled language. Therefore, it doesn't need a compiler to compile code into an executable program. This makes running an R script much less time-consuming.
- The number of R packages available either through CRAN or GitHub is 1,00,000 and they do epic stuff with just one line of code. It could range from Regression to Bayesian analysis.
- R is growing faster than any other data science language. It's the most-used data science language after SQL. It is used by 70% of data miners.
- One of the packages in R namely rmarkdown package helps you create reproducible Word documents and reproducible Powerpoint Presentations from your R markdown code just by changing one line in the YAML! ("YAML Ain't Markup Language!")
- It is really very easy in R to connect to almost any database using the dbplyer package. This makes it possible for an R user to work independently and pulling data from almost all common database types. You can also use packages like bigrquery to work directly with BigQuery and other high-performance data stores.
- You can build and host interactive web apps in just a few lines of code in R. Using the flexdashboard package in R you can create interactive web apps with a few lines of code. And using the rconnect package you can also host your web apps on your own server or, even easier, host them on a cloud server.

- You can not only deploy web apps but also can make them into awesome video games in R. The nesy package helps you create NES(The Nintendo Entertainment System) looking Shiny apps and deploy them just like you would any other Shiny app.
- You can build APIs and serve them from R. The plumber package in R helps you convert R functions to web APIs that can be integrated into downstream applications.

Environment in R

- The environment is a virtual space that is triggered when an interpreter of a programming language is launched.
- Simply, the environment is a collection of all the objects, variables, and functions. Or Environment can be assumed as a top-level object that contains the set of names/variables associated with some values.

Introduction to R studio

R Studio is an integrated development environment(IDE) for R. IDE is a GUI, where you can write your codes, see the results, and see the variables that are generated during the course of programming. R Studio is available as both Open source and Commercial software. R Studio is also available as both Desktop and Server versions. R Studio is also available for various platforms such as Windows, Linux, and macOS. R Studio is an open-source tool that provides an IDE to use R language, and enterprise-ready professional software for data science teams to develop and share the work with their team. R Studio can be downloaded from its official website (<https://rstudio.com/>)

13.1 Data Types

Variables are nothing but reserved memory locations to store values. This means that, when you create a variable, you reserve some space in memory. You may like to store information of various data types like character, wide character, integer, floating point, double floating point, Boolean etc. Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory. In contrast to other programming languages like C and Java in R, the variables are not declared as some data type. The variables are assigned with R-Objects and the data type of the R-object becomes the data type of the variable. There are many types of R-objects.

Various types of Data Types

- Vectors
- Lists
- Matrices
- Arrays
- Factors
- Data Frames

Vector object

The simplest of these objects is the **vector object** and there are six data types of these atomic vectors, also termed as six classes of vectors.

- 1) Logical – True, False
- 2) Numeric – 12.3, 5, 999
- 3) Integer – 2L, 3L, 0L
- 4) Complex – 3+2i
- 5) Character – 'a', "good", "TRUE", '23.4'

6) Raw - "Hello" is stored as 48 65 6c 6c 66

When you want to create vector with more than one element, you should use `c()` function which means to combine the elements into a vector.

- # Create a vector

```
apple <- c('red', 'apple', 'yellow')
print(apple)
```

- # Get the class of a vector

```
print(class(apple))
```

Lists

A list is an R-object which can contain many different types of elements inside it like vectors, functions and even another list inside it.

- #Create a list

```
list1 <- list(c(2,5,3),21.3,sin)
```

- #Print the list

```
Print(list1)
```

Matrix

A matrix is a two-dimensional rectangular data set. It can be created using a vector input to the `matrix` function.

- #Create a matrix

```
M = matrix(c('a', 'a', 'b', 'c', 'b', 'a'), nrow=2, ncol=3, byrow=TRUE)
```

```
Print(M)
```

Arrays

While matrices are confined to two dimensions, arrays can be of any number of dimensions. The `array` function takes a `dim` attribute which creates the required number of dimensions.

- #Create an array

```
a <- array(c('green', 'yellow'), dim=c(3,3,2))
```

```
print(a)
```

Factors

Factors are the R-objects which are created using a vector. It stores the vector along with the distinct values of the elements in the vector as labels. The labels are always character irrespective of whether it is numeric or character or Boolean etc. in the input vector. They are useful in statistical modeling. Factors are created using the `factor()` function. The `nlevels` function gives the count of levels.

- #Create a vector

```
apple_colors <- c('green', 'green', 'yellow', 'red', 'red', 'red', 'green')
```

- #Create a factor object

```
factor_apple <- factor(apple_colors)
```

- Print the factor

```
print(factor_apple)
print(nlevels(factor_apple))
```

Data Frames

Data frames are tabular data objects. Unlike a matrix in data frame each column can contain different modes of data. The first column can be numeric while the second column can be character and third column can be logical. It is a list of vectors of equal length. Data Frames are created using the **data.frame()** function.

- #Create the data frame

```
BMI <- data.frame(gender = c("Male", "Male", "Female"),
                 height = c(152, 171.5, 165),
                 weight = c(81, 93, 78),
                 age = c(42, 38, 26) )

print(BMI)
```

13.2 Variables

A variable provides us with named storage that our programs can manipulate. A variable in R can store an atomic vector, group of atomic vectors or a combination of many R objects. A valid variable name consists of letters, numbers and the dot or underline characters. The variable name starts with a letter, or the dot not followed by a number.

Valid variable names

- var_name2.
- .var_name,
- var.name

Invalid variable names

- var_name%
- 2var_name
- .2var_name
- _var_name

Variable assignment

The variables can be assigned values using leftward, rightward, and equal to operator. The values of the variables can be printed using **print()** or **cat()** function. The **cat()** function combines multiple items into a continuous print output.

Assignment with equal operator

- var.1 = c(0,1,2,3)

Assignment with leftward operator

- var.2 <- c("learn", "R")

Assignment with rightward operator

- C(TRUE,1) -> var.3

Printing of variables

```
cat("var.1 is ", var.1, "\n")
cat("var.2 is ", var.2, "\n")
cat("var.3 is ", var.3, "\n")
```

Data type of a variable

In R, a variable itself is not declared of any data type, rather it gets the data type of the R - object assigned to it. So, R is called a dynamically typed language, which means that we can change a variable's data type of the same variable again and again when using it in a program.

```
var_x<- "Hello"
cat("The class of var_x is ",class(var_x),"\n")
var_x<- 34.5
cat(" Now the class of var_x is ",class(var_x),"\n")
var_x<- 27L
cat(" Next the class of var_x becomes ",class(var_x),"\n")
```

Finding variables

To know all the variables currently available in the workspace we use the **ls()** function. Also, the **ls()** function can use patterns to match the variable names.

- Print(ls())

Deleting variables

Variables can be deleted by using the **rm()** function. Below we delete the variable var.3. On printing the value of the variable error is thrown.

```
rm(var.3)
print(var.3)
```

13.3 R operators

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. R language is rich in built-in operators and provides the following types of operators.

Types of operators

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Assignment Operators
- Miscellaneous Operators

Arithmetic operators

- + Addition

- - Subtraction
- * Multiplication
- / Division
- %% Remainder
- ^ Power

Relational Operators

- > Greater than
- < Less than
- == Equal to
- <= Less than or equal to
- >= Greater than or equal to
- != Not equal to

Logical Operators

- & Element wise Logical AND
- | Element wise Logical OR
- ! Element wise Logical NOT
- && Logical AND
- || Logical OR

Assignment operators

- <- or == or <<- Left assignment
- -> or ->> Right assignment

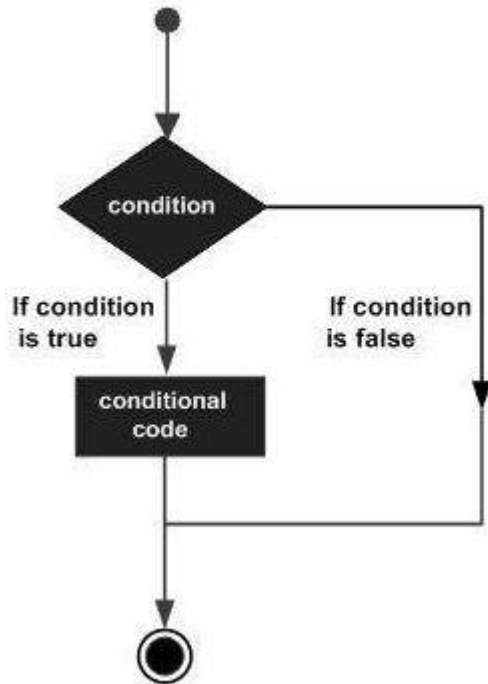
Miscellaneous operator

- : Colon operator
- %in% Identify an element belongs to a vector
- %*% Multiply a matrix with its transpose

13.4 Decision Making

Decision making structures require the programmer to specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be **true**, and optionally, other statements to be executed if the condition is determined to be **false**.

General form



Types of decision-making statements

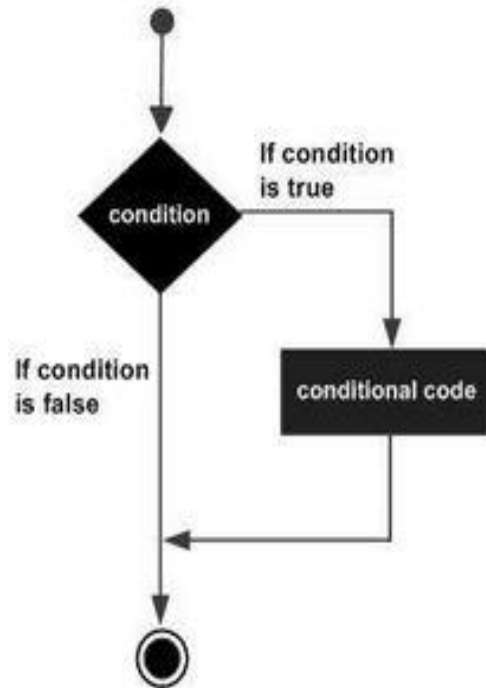
- If statement
- If.....else statement
- Switch statement

If statement

An **if** statement consists of a Boolean expression followed by one or more statements. The syntax is:

```
if(boolean_expression) {  
    // statement(s) will execute if the boolean expression is true.  
}
```

If the Boolean expression is evaluated to be **true**, then the block of code inside the if statement will be executed. If Boolean expression evaluates to be **false**, then the first set of code after the end of the if statement (after the closing curly brace) will be executed.



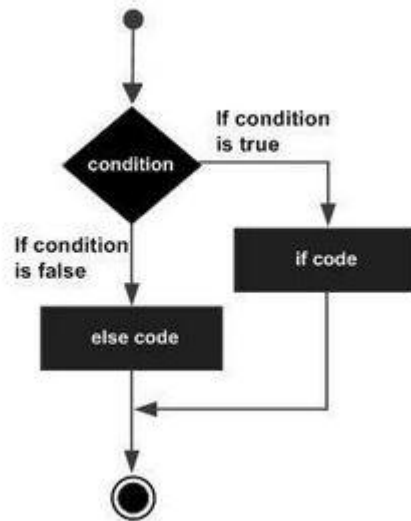
```
x <- 30L
if(is.integer(x)) {
  print("X is an Integer")
}
Output: [1] "X is an Integer"
```

If...Else statement

An **if** statement can be followed by an optional **else** statement which executes when the boolean expression is false. The syntax is:

```
if(boolean_expression) {
  // statement(s) will execute if the boolean expression is true.
} else {
  // statement(s) will execute if the boolean expression is false.
}
```

If the Boolean expression evaluates to be **true**, then the **if block** of code will be executed, otherwise **else block** of code will be executed.



```

x <- c("what","is","truth")
if("Truth" %in% x) {
  print("Truth is found")
} else {
  print("Truth is not found")
}
O/P: [1] "Truth is not found"

```

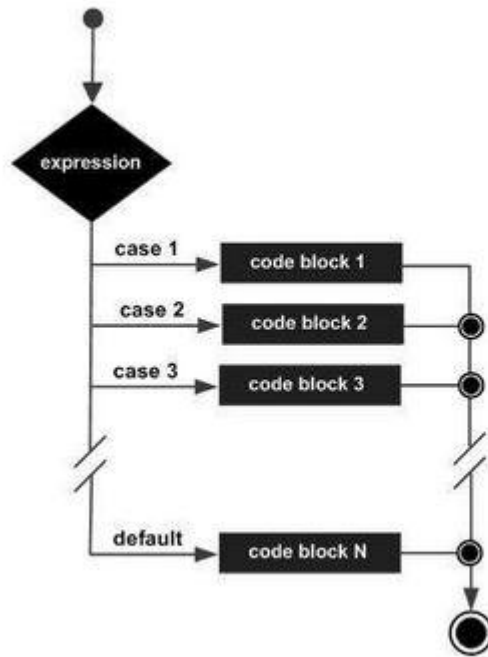
Switch statement

A **switch** statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case. The syntax is:

```
switch(expression, case1, case2, case3....)
```

Rules apply to a switch statement –

- If the value of expression is not a character string it is coerced to integer.
- You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.
- If the value of the integer is between 1 and `nargs()-1` (The max number of arguments) then the corresponding element of case condition is evaluated, and the result returned.
- If expression evaluates to a character string, then that string is matched (exactly) to the names of the elements.
- If there is more than one match, the first matching element is returned.
- No Default argument is available.
- In the case of no match, if there is an unnamed element of ... its value is returned. (If there is more than one such argument an error is returned.)

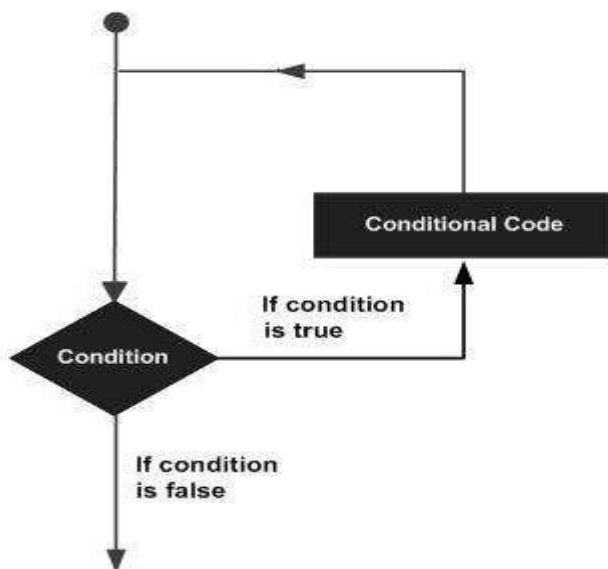


```

x <- switch(3, "first", "second", "third", "fourth")
print(x)
O/P: [1] "third"
  
```

13.5 Loops

There may be a situation when you need to execute a block of code several number of times. In general, statements are executed sequentially. The first statement in a function is executed first, followed by the second, and so on. Programming languages provide various control structures that allow for more complicated execution paths. A loop statement allows us to execute a statement or group of statements multiple times and the following is the general form of a loop statement in most of the programming languages.



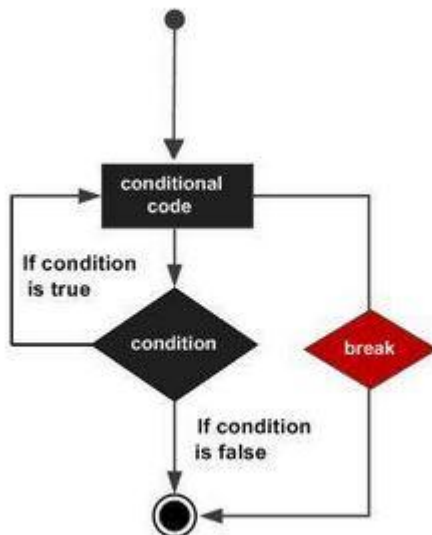
Kinds of Loops

- Repeat loop
- While loop
- For loop

Repeat Loop

The **Repeat loop** executes the same code again and again until a stop condition is met. The syntax is:

```
repeat {
  commands
  if(condition) {
    break
  }
}
```



```
v <- c("Hello","loop")
cnt<- 2
repeat {
  print(v)
  cnt<- cnt+1
  if(cnt> 5) {
    break
  }
}
```

O/P: [1] "Hello" "loop"

[1] "Hello" "loop"

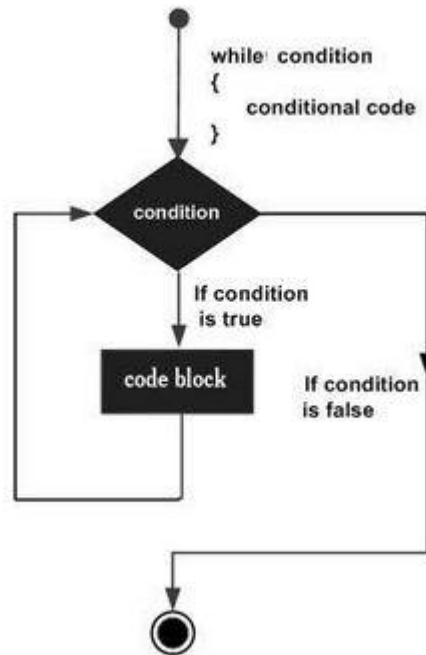
```
[1] "Hello" "loop"
```

```
[1] "Hello" "loop"
```

While Loop

The While loop executes the same code again and again until a stop condition is met. The syntax is:

```
while (test_expression) {
  statement
}
```



Here key point of the **while** loop is that the loop might not ever run. When the condition is tested and the result is false, the loop body will be skipped and the first statement after the while loop will be executed.

```
v <- c("Hello","while loop")
cnt<- 2
while (cnt< 7) {
  print(v)
  cnt = cnt + 1
}
```

```
[1] "Hello" "while loop"
```

```
[1] "Hello" "while loop"
```

```
[1] "Hello" "while loop"
```

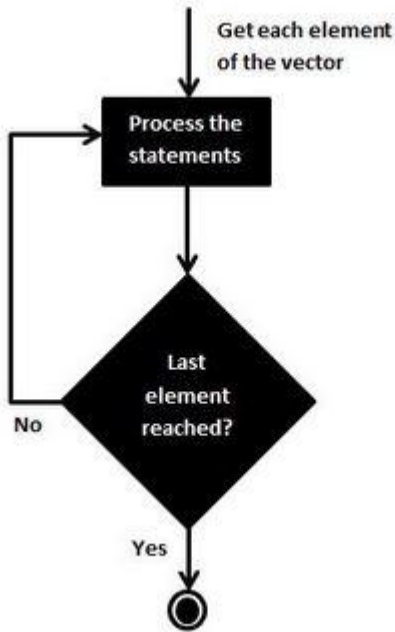
```
[1] "Hello" "while loop"
```

```
[1] "Hello" "while loop"
```

For loop

A **For loop** is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

```
for (value in vector) {
  statements
}
```



R's for loops are particularly flexible in that they are not limited to integers, or even numbers in the input. We can pass character vectors, logical vectors, lists or expressions.

```
v <- LETTERS[1:4]
for (i in v) {
  print(i)
}
```

O/P: [1] "A"

[1] "B"

[1] "C"

[1] "D"

13.6 Loop Control Statements

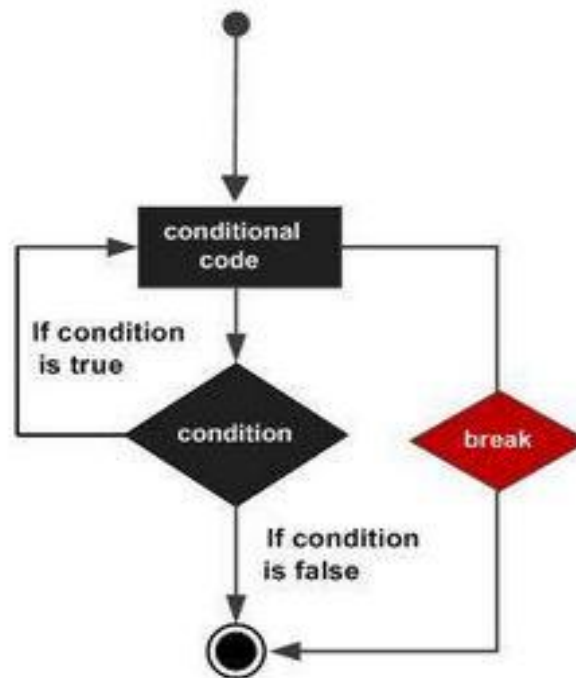
Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed. The loop control statements are:

- Break statement
- Next statement

Break Statement

The break statement in R programming language has the following two usages.

- 1) When the break statement is encountered inside a loop, the loop is immediately terminated, and program control resumes at the next statement following the loop.
- 2) It can be used to terminate a case in the switch statement.



```

v <- c("Hello","loop")
cnt<- 2
repeat {
  print(v)
  cnt<- cnt + 1
  if(cnt> 5) {
    break
  }
}
  
```

O/P: [1] "Hello" "loop"

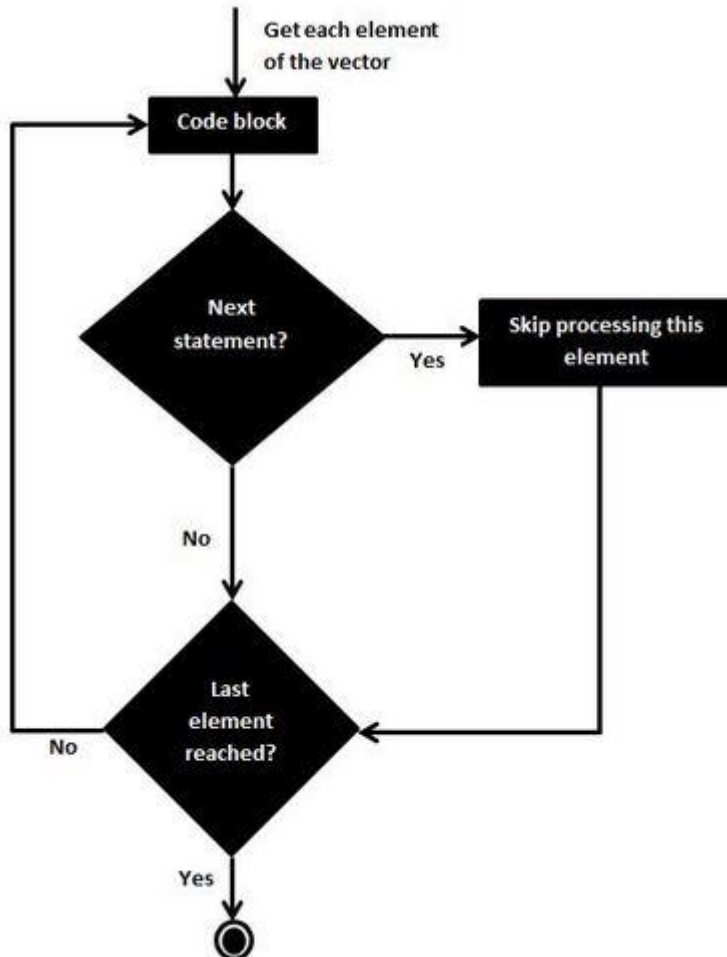
[1] "Hello" "loop"

[1] "Hello" "loop"

[1] "Hello" "loop"

Next Statement

The **next** statement in R programming language is useful when we want to skip the current iteration of a loop without terminating it. On encountering next, the R parser skips further evaluation and starts the next iteration of the loop.



```

v <- LETTERS[1:6]
for (i in v) {
  if (i == "D") {
    next
  }
  print(i)
}

```

O/P: [1] "A"
 [1] "B"
 [1] "C"
 [1] "E"

```
[1] "F"
```

13.7 Functions

A function is a set of statements organized together to perform a specific task. R has many in-built functions, and the user can create their own functions. In R, a function is an object so the R interpreter can pass control to the function, along with arguments that may be necessary for the function to accomplish the actions. The function in turn performs its task and returns control to the interpreter as well as any result which may be stored in other objects.

Function Definition

An R function is created by using the keyword **function**. The basic syntax of an R function definition is:

```
function_name <- function(arg_1, arg_2, ...) {
  function body
}
```

Function Components

- **Function Name** – This is the actual name of the function. It is stored in an R environment as an object with this name.
- **Arguments** – An argument is a placeholder. When a function is invoked, you pass a value to the argument. Arguments are optional; that is, a function may contain no arguments. Also, arguments can have default values.
- **Function Body** – The function body contains a collection of statements that defines what the function does.
- **Return Value** – The return value of a function is the last expression in the function body to be evaluated

R has many **in-built** functions which can be directly called in the program without defining them first. We can also create and use our own functions referred to as **user defined** functions.

Built-in Functions

The simple examples of in-built functions are `seq()`, `mean()`, `max()`, `sum(x)` and `paste(...)` etc. They are directly called by user written programs.

- # Create a sequence of numbers from 32 to 44.

```
print(seq(32,44))
```

- # Find mean of numbers from 25 to 82.

```
print(mean(25:82))
```

- # Find sum of numbers from 41 to 68.

```
print(sum(41:68))
```


O/P: [1] 32 33 34 35 36 37 38 39 40 41 42 43 44

[1] 53.5

[1] 1526

User Defined Functions

We can create user-defined functions in R. They are specific to what a user wants and once created they can be used like the built-in functions.

- # Create a function to print squares of numbers in sequence.

```
new.function<- function(a) {
  for(i in 1:a) {
    b <- i^2
    print(b)
  }
}
```

- Calling a function

Call the function new.function supplying 6 as an argument.

```
new.function(6)
```

O/P: [1] 1

[1] 4

[1] 9

[1] 16

[1] 25

[1] 36

- Calling a function without an argument

Create a function without an argument.

```
new.function<- function() {
  for(i in 1:5) {
    print(i^2)
  }
}
```

- # Call the function without supplying an argument.

```
new.function()
```

O/P: [1] 1

[1] 4

[1] 9

[1] 16

[1] 25

The arguments to a function call can be supplied in the same sequence as defined in the function or they can be supplied in a different sequence but assigned to the names of the arguments.

- # Create a function with arguments.

```
new.function<- function(a,b,c) {
  result <- a * b + c
  print(result)
}
```

- # Call the function by position of arguments.

```
new.function(5,3,11)
```

- # Call the function by names of the arguments.

```
new.function(a = 11, b = 5, c = 3)
```

- Calling a function with default arguments

We can define the value of the arguments in the function definition and call the function without supplying any argument to get the default result. But we can also call such functions by supplying new values of the argument and get non default result.

- # Create a function with arguments.

```
new.function<- function(a = 3, b = 6) {
  result <- a * b
  print(result)
}
```

- # Call the function without giving any argument.

```
new.function()
```

- # Call the function giving new values of the argument.

```
new.function(9,5)
```

O/P: [1] 18

[1] 45

13.8 Strings

Any value written within a pair of single quotes or double quotes in R is treated as a string. Internally R stores every string within double quotes, even when you create them with single quote.

Rules applied in string construction

- The quotes at the beginning and end of a string should be both double quotes or both single quote. They cannot be mixed.
- Double quotes can be inserted into a string starting and ending with single quote.
- Single quotes can be inserted into a string starting and ending with double quotes.
- Double quotes cannot be inserted into a string starting and ending with double quotes.
- Single quotes cannot be inserted into a string starting and ending with single quote.

Valid strings

```
a <- 'Start and end with single quote'
print(a)
b <- "Start and end with double quotes"
print(b)
c <- "single quote ' in between double quotes"
print(c)
d <- 'Double quotes " in between single quote'
print(d)
O/P: [1] "Start and end with single quote"
[1] "Start and end with double quotes"
[1] "single quote ' in between double quote"
[1] "Double quote \" in between single quote"
```

Invalid strings

```
e <- 'Mixed quotes"
print(e)
f <- 'Single quote ' inside single quote'
print(f)
g <- "Double quotes " inside double quotes"
```

```
print(g)
Error: unexpected symbol in:
"print(e)
f <- 'Single"
Execution halted
```

String manipulation

- Concatenating Strings - paste () function
- Formatting numbers & strings - format() function
- Counting number of characters in a string - nchar() function
- Changing the case - toupper() & tolower() functions
- Extracting parts of a string - substring () function

Concatenating Strings - paste() function

Many strings in R are combined using the **paste()** function. It can take any number of arguments to be combined.

```
paste (... , sep = " ", collapse = NULL)
```

- ... represents any number of arguments to be combined.
- **sep** represents any separator between the arguments. It is optional.
- **collapse** is used to eliminate the space in between two strings. But not the space within two words of one string.

```
a <- "Hello"
b <- 'How'
c <- "are you? "
print(paste(a,b,c))
print(paste(a,b,c, sep = "-"))
print(paste(a,b,c, sep = "", collapse = ""))
```

```
O/P: [1] "Hello How are you? "
```

```
[1] "Hello-How-are you? "
```

```
[1] "HelloHoware you? "
```

Formatting numbers & strings - format() function

Numbers and strings can be formatted to a specific style using **format()** function.

```
format(x, digits, nsmall, scientific, width, justify = c("left", "right", "centre", "none"))
```

- **x** is the vector input.

- **digits** are the total number of digits displayed.
- **nsmall** is the minimum number of digits to the right of the decimal point.
- **scientific** is set to TRUE to display scientific notation.
- **width** indicates the minimum width to be displayed by padding blanks in the beginning.
- **Justify** is the display of the string to left, right or center.

Total number of digits displayed. Last digit rounded off.

```
result <- format(23.123456789, digits = 9)
```

```
print(result)
```

- # Display numbers in scientific notation.

```
result <- format(c(6, 13.14521), scientific = TRUE)
```

```
print(result)
```

- # The minimum number of digits to the right of the decimal point.

```
result <- format(23.47, nsmall = 5)
```

```
print(result)
```

- # Format treats everything as a string.

```
result <- format(6)
```

```
print(result)
```

- # Numbers are padded with blank in the beginning for width.

```
result <- format(13.7, width = 6)
```

```
print(result)
```

- # Left justify strings.

```
result <- format("Hello", width = 8, justify = "l")
```

```
print(result)
```

- # Justify string with center.

```
result <- format("Hello", width = 8, justify = "c")
```

```
print(result)
```

```
O/P: [1] "23.1234568"
```

```
[1] "6.000000e+00" "1.314521e+01"
```

```
[1] "23.47000"
```

```
[1] "6"
```

```
[1] " 13.7"
```

```
[1] "Hello "
```

```
[1] " Hello "
```

Counting number of characters in a string - nchar() function

This function counts the number of characters including spaces in a string. The syntax is:

```
nchar(x)
```

x is the vector input.

```
result <- nchar("Count the number of characters")
```

```
print(result)
```

```
O/P: [1] 30
```

Changing the case - toupper() & tolower() functions

These functions change the case of characters of a string.

```
toupper(x)
```

```
tolower(x)
```

- x is the vector input.
- # Changing to Upper case.

```
result <- toupper("Changing to Upper")
```

```
print(result)
```

- # Changing to lower case.

```
result <- tolower("Changing to Lower")
```

```
print(result)
```

- O/P: [1] "CHANGING TO UPPER"

```
[1] "changing to lower"
```

Extracting parts of a string - substring() function

This function extracts parts of a String.

- substring(x,first,last)

Following is the description of the parameters used –

- x is the character vector input.
- first is the position of the first character to be extracted.
- last is the position of the last character to be extracted
- # Extract characters from 5th to 7th position.

```
result <- substring("Extract", 5, 7)
```

```
print(result)
```

```
O/P: [1] "act"
```

13.9 R Packages

R packages are a collection of R functions, compiled code and sample data. They are stored under a directory called "**library**" in the R environment. By default, R installs a set of packages during installation. More packages are added later when they are needed for some specific purpose. When we start the R console, only the default packages are available by default. Other packages which are already installed must be loaded explicitly to be used by the R program that is going to use them.

Check available R packages

- Get library locations containing R packages: `.libPaths()`
- Get the list of all the packages installed: `library()`

Install a new package

There are two ways to add new R packages. One is installing directly from the CRAN directory, and another is downloading the package to your local system and installing it manually.

Install from CRAN

```
install.packages("Package Name")
```

```
# Install the package named "XML".
```

```
install.packages("XML")
```

Install package manually

```
install.packages(file_name_with_path, repos = NULL, type = "source")
```

```
# Install the package named "XML"
```

```
install.packages("E:/XML_3.98-1.3.zip", repos = NULL, type = "source")
```

Load package to library

Before a package can be used in the code, it must be loaded to the current R environment. You also need to load a package that is already installed previously but not available in the current environment. A package is loaded using the following command –

```
library ("package Name", lib.loc = "path to library")
```

```
# Load the package named "XML"
```

```
install.packages("E:/XML_3.98-1.3.zip", repos = NULL, type = "source")
```

13.10 Data Reshaping

Data Reshaping in R is about changing the way data is organized into rows and columns. Most of the time data processing in R is done by taking the input data as a data frame. Data Reshaping in R is about changing the way data is organized into rows and columns. Most of the time data processing in R is done by taking the input data as a data frame.

Joining Columns and Rows in a Data Frame

We can join multiple vectors to create a data frame using the `cbind()` function. Also, we can merge two data frames using `rbind()` function.

Merging data frames

We can merge two data frames by using the `merge()` function. The data frames must have the same column names on which the merging happens.

Melting and casting

One of the most interesting aspects of R programming is about changing the shape of the data in multiple steps to get a desired shape. The functions used to do this are called `melt()` and `cast()`.

Summary

- R is an open-source programming language mostly used for statistical computing and data analysis. It is available across widely used platforms like Windows, Linux, and MacOS.
- R programming is used as a leading tool for machine learning, statistics, and data analysis. Objects, functions, and packages can easily be created by R.
- As R programming language is an open source. Thus, you can run R anywhere and at any time.
- Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory.
- The data types in R are: vectors, lists, matrices, arrays, factors and data frames.
- Factors are the R-objects which are created using a vector. It stores the vector along with the distinct values of the elements in the vector as labels.
- A valid variable name consists of letters, numbers and the dot or underline characters.
- The variables can be assigned values using leftward, rightward and equal to operator.
- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. R language is rich in built-in operators and provides the following types of operators.
- A function is a set of statements organized together to perform a specific task. R has many in-built functions, and the user can create their own functions.

Keywords

- **R:** R is an interpreted language that supports both procedural programming and object-oriented programming. This is an implementation of the S programming language.
- **RStudio:** R Studio is an integrated development environment (IDE) for R. IDE is a GUI, where you can write your quotes, see the results, and also see the variables that are generated during the course of programming.

- **R Objects:** The variables are assigned with R-Objects and the data type of the R-object becomes the data type of the variable. There are many types of R-objects.
- **Lists:** A list is an R-object which can contain many different types of elements inside it like vectors, functions and even another list inside it.
- **Variable in R:** A variable provides us with named storage that our programs can manipulate. A variable in R can store an atomic vector, group of atomic vectors or a combination of many Robjects.
- **Loops:** A loop statement allows us to execute a statement or group of statements multiple times and the following is the general form of a loop statement in most of the programming languages.

SelfAssessment

1. R language is used for
 - A. Statistical computing
 - B. Data analysis
 - C. Fixing the bugs
 - D. All of the above

2. R language is available across
 - A. Windows
 - B. MAC
 - C. Linux
 - D. All of the above

3. R is an interpreted language which supports
 - A. Only procedural programming
 - B. Object-oriented programming
 - C. Both procedural and object-oriented programming
 - D. None of the above

4. Which of the following are the data types in R programming language?
 - A. Vectors
 - B. Lists
 - C. Data frames
 - D. All of the above mentioned

5. Which attribute creates the required number of dimensions in arrays?
 - A. dimension
 - B. dim
 - C. sides
 - D. All of the above

6. The values of the variables can be printed using
 - A. Cat()
 - B. Print()
 - C. Both above mentioned
 - D. None of the above

7. The decision-making structures in R programming languages are
 - A. If statement
 - B. If-else statement
 - C. Switch statement
 - D. All the above

8. In switch statement, the number of cases allowed are:
 - A. 1
 - B. 2
 - C. 3
 - D. Any

9. Each case in switch statement is followed by the value to be compared to and a
 - A. Colon
 - B. Comma
 - C. Semi-colon
 - D. Asterisk

10. The kinds of loops in R language are
 - A. While loop
 - B. Repeat loop
 - C. For loop
 - D. All of the above

11. Loop control statements in R are
 - A. Break statement
 - B. Next statement
 - C. Both of the above
 - D. None of the above

12. Which of the following are the function components?
 - A. Function body
 - B. Function name
 - C. Arguments
 - D. All of the above mentioned

13. The strings in R programming language can be written in
 - A. Single quotes
 - B. Double quotes
 - C. Either single or double quotes
 - D. None of the above

14. The concatenation of strings can be done using
 - A. paste()
 - B. format()
 - C. nchar()
 - D. substring()

15. The numbers and strings can be formatted to a specific style using
 - A. paste()
 - B. format()
 - C. nchar()
 - D. substring()

Answers for Self Assessment

- | | | | | |
|-------|-------|-------|-------|-------|
| 1. D | 2. D | 3. C | 4. D | 5. B |
| 6. D | 7. D | 8. D | 9. A | 10. D |
| 11. C | 12. D | 13. D | 14. A | 15. B |

Review Questions

1. Why is R programming language used? Also explain the features of R programming language.
2. What are the advantages and disadvantages of R programming language?
3. What is a data type? Which data types exist in R programming language?
4. What is a vector object? How do we create a vector, and get the class of a vector?
5. What are operators? Explain its types.
6. What is decision making structures in R programming language? Explain.

**Further Readings**

<https://www.r-project.org/about.html>

<https://www.geeksforgeeks.org/r-programming-language-introduction/>

Unit 14: NumPy and Pandas

CONTENTS

Objectives

Introduction

- 14.1 Python
- 14.2 First Python Program
- 14.3 Python Variables
- 14.4 Python Data Types
- 14.5 Lists
- 14.6 Dictionaries
- 14.7 Tuples
- 14.8 Files
- 14.9 Other Core Data Types
- 14.10 NumPy
- 14.11 Operations on NumPy Arrays
- 14.12 Data Types in NumPy
- 14.13 Creating Arrays
- 14.14 NumPy Operations
- 14.15 NumPy Array Shape
- 14.16 Reshaping NumPy arrays
- 14.17 NumPy Array Iterating
- 14.18 Joining NumPy Arrays
- 14.19 NumPy Splitting Arrays
- 14.20 NumPy Array Search
- 14.21 NumPy Sorting arrays
- 14.22 NumPy Filter Arrays
- 14.23 Random Number in NumPy
- 14.24 Pandas
- 14.25 Why Pandas?
- 14.26 Installing and Importing Pandas
- 14.27 Data Structures of Pandas
- 14.28 Data Cleaning
- 14.29 Data Transformation Operations

Summary

Keywords

Self Assessment

Answers for Self Assessment

Review Questions

Further Readings

Objectives

After studying this unit, you will be able to:

- Understand the basics of Python
- Understand data types in python
- Understand NumPy and its data types
- Understand different NumPy operations
- Understand NumPy sorting and filter arrays
- Understand random numbers in NumPy
- Understand the basic concept of pandas and its data structures
- Understand how to clean the data and various preprocessing operations

Introduction

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. These are the features of Python programming language.

14.1 Python

- **Interpreted** - An interpreted language is a programming language which are generally interpreted, without compiling a program into machine instructions. It is one where the instructions are not directly executed by the target machine, but instead read and executed by some other program.
- **Object-oriented** - Object-oriented language is a high-level computer programming language that implements objects and their associated procedures within the programming context to create software programs. Object-oriented language uses an object-oriented programming technique that binds related data and functions into an object and encourages reuse of these objects within the same and other programs.
- **High-level** - Python is a high-level language which just means that it's simpler for a human to use. Low-level languages such as C/C++ require a much more detailed understanding of how a computer works. With a high-level language, many of these details are abstracted away to make your life easier. Python uses dynamic semantics, meaning that its variables are dynamic objects.

Few Other Features of Python are

1. Popular
2. User-friendly
3. Simple
4. Highly powerful
5. Open source
6. General purpose

Comparison with other language

Python is often compared to other interpreted languages such as Java, JavaScript, Perl, Tcl, or Smalltalk.

1. Java - Python programs are generally expected to run slower than Java programs, but they also take much less time to develop. Python programs are typically 3-5 times shorter than equivalent Java programs. This difference can be attributed to Python's built-in high-level data types and its dynamic typing.
2. Javascript - Python's "object-based" subset is roughly equivalent to JavaScript. Like JavaScript (and unlike Java), Python supports a programming style that uses simple functions and variables without engaging in class definitions.
3. Perl - Perl emphasizes support for common application-oriented tasks, e.g., by having built-in regular expressions, file scanning and report generating features. Python emphasizes support for common programming methodologies such as data structure design and object-oriented programming and encourages programmers to write readable (and thus maintainable) code by providing an elegant but not overly cryptic notation.
4. C++ - Python code is typically 3-5 times shorter than equivalent Java code; it is often 5-10 times shorter than equivalent C++ code.

Uses of Python

Python is a very versatile language. Its uses are:

1. Web applications: Popular frameworks like the Django web application and Flask are written in Python.
2. Desktop applications: The Dropbox client is written in Python.
3. Scientific and numeric computing: Python is the top choice for data science and machine learning.
4. Cybersecurity: Python is excellent for data analysis, writing system scripts that interact with an operating system, and communicating over network sockets.

What can Python do?

1. Python can be used on a server to create web applications.
2. Python can be used alongside software to create workflows.
3. Python can connect to database systems. It can also read and modify files.
4. Python can be used to handle big data and perform complex mathematics.
5. Python can be used for rapid prototyping, or for production-ready software development.

Why to study Python?

1. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
2. Python has a simple syntax like the English language.
3. Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
4. Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
5. Python can be treated in a procedural way, an object-oriented way or a functional way.

Download and install Python

- Open browser
- Open python.org
- Click on downloads
- Download the latest version

Download a Code Editor

- PyCharm
- www.jetbrains.com/pycharm

- Download pycharm
- Download Community edition (free)

14.2 First Python Program

- Open the project.
- Right click on it.
- Create a new python file.
- Save it with extension .py.
- Write the program.
 - Printing anything on screen

```
print('Data Science Toolbox')
```

OR

```
print("Data Science Toolbox")
```

- For printing a statement, single or double quotation marks can be used to print anything.

How to run?

1. Go to run →run.
2. Shortcut = Alt + Shift + F10.
3. See the result in the terminal window.

Python Indentation

Indentation refers to the spaces at the beginning of a code line. Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important. Python uses indentation to indicate a block of code. For example

```
if 5 > 2:
```

```
print("Five is greater than two!")
```

But this will show the syntax error

```
if 5 > 2:
```

```
print("Five is greater than two!")
```

Python comments

Comments can be used to explain Python code. Comments can be used to make the code more readable. Comments can be used to prevent execution when testing code. Comments can be created using a specific way in Python. Comments start with # and Python will ignore them. For example:

```
#This is a comment
```

For multiline comments, the Python does not have a syntax for multiline comments. To add a multi-line comment, you need to insert # in every line.

14.3 Python Variables

Variables are containers for storing data values. Python has no command for declaring a variable. A variable is created the moment you first assign a value to it. For example:

```
x = 5
```

```
y = "John"
```

```
print(x)
```

```
print(y)
```

Variables do not need to be declared with any particular type, and can even change type after they have been set. For example:

```
x = 4    # x is of type int
x = "Sally" # x is now of type str
print(x)
```

Type Casting

If you want to specify the data type of a variable, this can be done with casting. For example:

```
x = str(3)  # x will be '3'
y = int(3)  # y will be 3
z = float(3) # z will be 3.0
```

Getting the type of a variable

You can get the data type of a variable with the `type()` function.

```
x = 5
y = "John"
print(type(x))
print(type(y))
```

Declaration of variables

String variables can be declared either by using single or double quotes. For example:

```
x = "John"
# is the same as
x = 'John'
```

Case-sensitiveness

Variable names are case-sensitive in nature.

```
a = 4
A = "Sally"
#A will not overwrite a
```

Rules for variable names

A variable can have a short name (like `x` and `y`) or a more descriptive name (`age`, `carname`, `total_volume`). The rules for Python variables:

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and `_`)
- Variable names are case-sensitive (`age`, `Age` and `AGE` are three different variables)

Few Legal variable names are

```
myvar = "John"
my_var = "John"
_my_var = "John"
myVar = "John"
MYVAR = "John"
myvar2 = "John"
```

Few Illegal variable names are

```
2myvar = "John"
```



```
my-var = "John"
```

```
my var = "John"
```

Variable names with more than one word can be difficult to read. So, for multiword names, the declaration can be different types. These are:

1. Camel Case: Each word, except the first, starts with a capital letter:

```
myVariableName = "John"
```

2. Pascal Case: Each word starts with a capital letter:

```
MyVariableName = "John"
```

3. Snake case: Each word is separated by an underscore character:

```
my_variable_name = "John"
```

For assigning multiple values, there are different ways. These are

1. Many Values to Multiple Variables: Python allows you to assign values to multiple variables in one line:

```
x, y, z = "Orange", "Banana", "Cherry"
```

```
print(x)
```

```
print(y)
```

```
print(z)
```

2. One Value to Multiple Variables: You can assign the same value to multiple variables in one line:

```
x = y = z = "Orange"
```

```
print(x)
```

```
print(y)
```

```
print(z)
```

3. Unpack a Collection: If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called unpacking.

```
fruits = ["apple", "banana", "cherry"]
```

```
x, y, z = fruits
```

```
print(x)
```

```
print(y)
```

```
print(z)
```

Output variables

The python print () function is used to output variables.

- x = "Python is awesome"

```
print(x)
```

In the print () function, you can output multiple variables separated by a comma

- x = "Python"

```
y = "is"
```

```
z = "awesome"
```

```
print (x, y, z)
```

You can also use + operator to output multiple variables.

```

• x = "Python "
  y = "is "
  z = "awesome"
  print (x + y + z)

```

14.4 Python Data Types

In all programming languages, data types are used to classify one particular type of data. This is important because the specific data type you use will determine what values you can assign to it and what you can do to it (including what operations you can perform on it).

Numbers

Python's core objects set includes the usual suspects:

- Integers (numbers without a fractional part),
- Floating-point numbers (roughly, numbers with a decimal point in them),
- Exotic numeric types (complex numbers with imaginary parts, fixed-precision decimals, rational fractions with numerator and denominator, and full featured sets).

Numbers in Python support normal mathematical operations. For instance, the plus sign (+) performs addition, a star (*) is used for multiplication, and two stars (**) are used for exponentiation:

```

• >>>123 + 222                                # Integer addition
  345

• >>>1.5 * 4                                  # Floating-point multiplication
  6.0

• >>>2 ** 100                                  # 2 to the power 100
  1267650600228229401496703205376

```

Besides expressions, there are a handful of useful numeric modules that ship with Python – modules are just packages of additional tools that we import to use:

```

>>> import math
>>> math.pi
3.1415926535897931
>>> math.sqrt(85)
9.2195444572928871

```

Random module performs random number generation and random selections

```

>>> import random
>>> random.random()
0.59268735266273953
>>> random.choice([1, 2, 3, 4])
1

```

Strings

Strings are used to record textual information as well as arbitrary collections of bytes. A sequence in Python – that is, a positionally ordered collection of other objects. Sequences maintain a left-to-right order among the items they contain: their items are stored and fetched by their relative position.

Sequence Operations

1. Indexing:

As sequences, strings support operations that assume a positional ordering among items. For example, if we have a four-character string, we can verify its length with the built-in len function and fetch its components with indexing expressions:

```
>>> S = 'Spam'
>>> len(S)                # Length
4
>>> S[0]                  # The first item in S, indexing by zero-based position
'S'
>>> S[1]                  # The second item from the left
'p'
```

In Python, indexes are coded as offsets from the front, and so start from 0: the first item is at index 0, the second is at index 1, and so on. Python variables never need to be declared ahead of time. A variable is created when you assign it a value, may be assigned any type of object, and is replaced with its value when it shows up in an expression. It must also have been previously assigned by the time you use its value. In Python, we can also index backward, from the end – positive indexes count from the left, and negative indexes count back from the right:

```
>>> S[-1]                # The last item from the end in S
'm'
>>> S[-2]                # The second to last item from the end
'a'
```

2. Slicing

In addition to simple positional indexing, sequences also support a more general form of indexing known as slicing, which is a way to extract an entire section (slice) in a single step. For example:

```
>>> S                    # A 4-character string
'Spam'
>>> S[1:3]               # Slice of S from offsets 1 through 2 (not 3)
'pa'
```

The easiest way to think of slices is that they are a way to extract an entire column from a string in a single step. Their general form, X [I: J], means “give me everything in X from offset I up to but not including offset J.” In a slice, the left bound defaults to zero, and the right bound defaults to the length of the sequence being sliced. This leads to some common usage variations:

```
>>> S[1:]                # Everything past the first (1:len(S))
'pam'
>>> S                    # S itself hasn't changed
'Spam'
>>> S[0:3]              # Everything but the last
'Spa'
>>> S[:3]               # Same as S[0:3]
'Spa'
>>> S[:-1]              # Everything but the last again, but simpler (0:-1)
'Spa'
>>> S[:]                 # All of S as a top-level copy (0:len(S))
'Spam'
```

3. Concatenation:

As sequences, strings also support concatenation with the plus sign (joining two strings into a new string) and repetition (making a new string by repeating another):

```
>>>S
'Spam'
>>>S + 'xyz'                # Concatenation
'Spamxyz'
>>>S                        # S is unchanged
'Spam'
>>>S * 8                    # Repetition
'SpamSpamSpamSpamSpamSpamSpamSpam'
```

4. Type specific methods

- Find

The string find method is the basic substring search operation (it returns the offset of the passed-in substring, or -1 if it is not present),

```
>>>S.find('pa')             # Find the offset of a substring
1
>>>S
'Spam'
```

- Replace

The string replace method performs global searches and replacements:

```
>>>S.replace('pa', 'XYZ')   # Replace occurrences of a substring with another
'SXYZm'
>>>S
'Spam'
```

Despite the names of these string methods, we are not changing the original strings here, but creating new strings as the results – because strings are immutable, we have to do it this way. String methods are the first line of text-processing tools in Python. Other methods split a string into substrings on a delimiter (handy as a simple form of parsing), perform case conversions, test the content of the string (digits, letters, and so on), and strip whitespace characters off the ends of the string:

- >>> line = 'aaa,bbb,cccc,dd'
 - >>>line.split(',') # Split on a delimiter into a list of substrings
 - ['aaa', 'bbb', 'cccc', 'dd']
 - >>> S = 'spam'
 - >>>S.upper() # Upper- and lowercase conversions
 - 'SPAM'
 - >>>S.isalpha() # Content tests: isalpha, isdigit, etc.
 - True
 - >>> line = 'aaa,bbb,cccc,dd\n'
 - >>> line = line.rstrip() # Remove whitespace characters on the right side
 - >>> line
- ```
'aaa,bbb,cccc,dd'
```

## 14.5 Lists

The Python list object is the most general sequence provided by the language. The lists are positionally ordered collections of arbitrarily typed objects, and they have no fixed size. They are also mutable—unlike strings, lists can be modified in-place by assignment to offsets as well as a variety of list method calls.

### Sequence operations

Because they are sequences, lists support all the sequence operations as strings; the only difference is that the results are usually lists instead of strings.

```
>>>L = [123, 'spam', 1.23] # A list of three different-type objects
>>>len(L) # Number of items in the list
3
>>>L[0] # Indexing by position
123
>>>L[:-1] # Slicing a list returns a new list
[123, 'spam']
>>>L + [4, 5, 6] # Concatenation makes a new list too
[123, 'spam', 1.23, 4, 5, 6]
>>>L # We're not changing the original list
[123, 'spam', 1.23]
```

### Type specific operations

Python's lists are related to arrays in other languages, but they tend to be more powerful. For one thing, they have no fixed type of constraint—the list we just looked at, for example, contains three objects of completely different types (an integer, a string, and a floating-point number). Further, lists have no fixed size. That is, they can grow and shrink on demand, in response to list-specific operations.

```
>>>L.append('NI') # Growing: add object at end of list
>>>L
[123, 'spam', 1.23, 'NI']
>>>L.pop(2) # Shrinking: delete an item in the middle
1.23
>>> L # "del L[2]" deletes from a list too
[123, 'spam', 'NI']
```

Because lists are mutable, most list methods also change the list object in-place, instead of creating a new one:

```
>>> M = ['bb', 'aa', 'cc']
>>>M.sort()
>>> M
['aa', 'bb', 'cc']
>>>M.reverse()
>>> M
['cc', 'bb', 'aa']
```

Although lists have no fixed size, Python still doesn't allow us to reference items that are not present. Indexing off the end of a list is always a mistake, but so is assigning off the end:

```
>>>L
```

```
[123, 'spam', 'NI']
>>>L [99]
...error text omitted...
IndexError: list index out of range
```

## Nesting

One nice feature of Python's core data types is that they support arbitrary nesting—we can nest them in any combination, and as deeply as we like. One immediate application of this feature is to represent matrices, or “multidimensional arrays” in Python. A list with nested lists will do the job for basic applications:

```
>>>M = [[1, 2, 3], # A 3 × 3 matrix, as nested lists
[4, 5, 6], # Code can span lines if bracketed
[7, 8, 9]]
>>>M
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

Here, we've coded a list that contains three other lists. The effect is to represent a  $3 \times 3$  matrix of numbers. Such a structure can be accessed in a variety of ways:

```
>>>M[1] # Get row 2
[4, 5, 6]
>>>M[1][2] # Get row 2, then get item 3 within the row
6
```

## 14.6 Dictionaries

Python dictionaries are something completely different—they are not sequences at all, but are instead known as mappings. Mappings are also collections of other objects, but they store objects by key instead of by relative position. In fact, mappings don't maintain any reliable left-to-right order; they simply map keys to associated values. Dictionaries, the only mapping type in Python's core objects set, are also mutable: they may be changed in-place and can grow and shrink on demand, like lists.

### Mapping Operations

1) When written as literals, dictionaries are coded in curly braces and consist of a series of “key: value” pairs. Dictionaries are useful anytime we need to associate a set of values with keys—to describe the properties of something, for instance. As an example, consider the following three-item dictionary (with keys “food,” “quantity,” and “color”):

```
>>> D = {'food': 'Spam', 'quantity': 4, 'color': 'pink'}
```

2) We can index this dictionary by key to fetch and change the keys' associated values. The dictionary index operation uses the same syntax as that used for sequences, but the item in the square brackets is a key, not a relative position:

```
>>>D['food'] # Fetch value of key 'food'
'Spam'
>>>D['quantity'] += 1 # Add 1 to 'quantity' value
>>>D
{'food': 'Spam', 'color': 'pink', 'quantity': 5}
```

The example starts with an empty dictionary and fills it out one key at a time. Unlike out-of-bounds assignments in lists, which are forbidden, assignments to new dictionary keys create those keys:

```
>>> D = {}
>>> D['name'] = 'Bob' # Create keys by assignment
```

```
>>> D['job'] = 'dev'
>>> D['age'] = 40
>>> D
{'age': 40, 'job': 'dev', 'name': 'Bob'}
>>> print(D['name'])
Bob
```

## 14.7 Tuples

The tuple object is roughly like a list that cannot be changed – tuples are sequences, like lists, but they are immutable, like strings. Syntactically, they are coded in parentheses instead of square brackets, and they support arbitrary types, arbitrary nesting, and the usual sequence operations.

```
>>> T = (1, 2, 3, 4) # A 4-item tuple
>>> len(T) # Length
4
>> T + (5, 6) # Concatenation
(1, 2, 3, 4, 5, 6)
>>> T[0] # Indexing, slicing, and more
1
```

### Type Specific Methods

Tuples also have two type-specific callable methods in Python 3.0, but not nearly as many as lists:

```
>>> T.index(4) # Tuple methods: 4 appears at offset 3
3
>>> T.count(4) # 4 appears once
1
```

The primary distinction for tuples is that they cannot be changed once created. That is, they are immutable sequences:

```
>>> T[0] = 2 # Tuples are immutable
...error text omitted...
```

TypeError: 'tuple' object does not support item assignment

Like lists and dictionaries, tuples support mixed types and nesting, but they don't grow and shrink because they are immutable:

```
>>> T = ('spam', 3.0, [11, 22, 33])
>>> T[1]
3.0
>>> T[2][1]
22
>>> T.append(4)
```

AttributeError: 'tuple' object has no attribute 'append'

Frankly, tuples are not generally used as often as lists in practice, but their immutability is the whole point. If you pass a collection of objects around your program as a list, it can be changed anywhere; if you use a tuple, it cannot. That is, tuples provide a sort of integrity constraint that is convenient in programs.

## 14.8 Files

File objects are Python code's main interface to external files on your computer. Files are a core type, but they're something of an oddball—there is no specific literal syntax for creating them. Rather, to create a file object, you call the built-in open function, passing in an external filename and a processing mode as strings. To create a text output file, you would pass in its name and the 'w' processing mode string to write data:

```
>>>f = open('data.txt', 'w') # Make a new file in output mode
>>>f.write('Hello\n') # Write strings of bytes to it
6
>>>f.write('world\n') # Returns number of bytes written in Python 3.0
6
>>>f.close() # Close to flush output buffers to disk
```

This creates a file in the current directory and writes text to it (the filename can be a full directory path if you need to access a file elsewhere on your computer). To read back what you just wrote, reopen the file in 'r' processing mode, for reading text input—this is the default if you omit the mode in the call. Then read the file's content into a string, and display it. A file's contents are always a string in your script, regardless of the type of data the file contains:

```
>>>f = open('data.txt') # 'r' is the default processing mode
>>>text = f.read() # Read entire file into a string
>>>text
'Hello\nworld\n'
>>>print(text) # print interprets control characters
Hello
world
>>>text.split() # File content is always a string
['Hello', 'world']
```

## 14.9 Other Core Data Types

- 1.Sets
- 2.Booleans

### 1. Sets

Sets are neither mappings nor sequences; rather, they are unordered collections of unique and immutable objects. Sets are created by calling the built-in set function or using new set literals and expressions in 3.0, and they support the usual mathematical set operations (the choice of new {...} syntax for set literals in 3.0 makes sense, since sets are much like the keys of a valueless dictionary).

```
>>>X = set('spam') # Make a set out of a sequence in 2.6 and 3.0
>>>Y = {'h', 'a', 'm'} # Make a set with new 3.0 set literals
>>>X, Y
({'a', 'p', 's', 'm'}, {'a', 'h', 'm'})
>>>X & Y # Intersection
{'a', 'm'}
>>>X | Y # Union
{'a', 'p', 's', 'h', 'm'}
>>>X - Y # Difference
```



```
{'p', 's'}
>>>{x ** 2 for x in [1, 2, 3, 4]} # Set comprehensions in 3.0
{16, 1, 4, 9}
```

## 2. Booleans

Python also comes with Booleans (with predefined True and False objects that are essentially just the integers 1 and 0 with custom display logic), and it has long supported a special placeholder object called None commonly used to initialize names and objects:

```
>>> 1 > 2, 1 < 2 # Booleans
(False, True)
>>> bool('spam')
True
>>> X = None # None placeholder
>>> print(X)
None
>>> L = [None] * 100 ` # Initialize a list of 100 Nones
>>> L
```

[None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, None, ...a list of 100 Nones...]

## 14.10 NumPy

It is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy is open-source software and has many contributors.

### Why use NumPy?

- In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.
- The array object in numpy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy.
- Arrays are very frequently used in data science, where speed and resources are very important.

### Why is numPy faster than lists?

NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behavior is called locality of reference.

### Language of numPy

NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.

### Installation of numpy

If you have Python and PIP already installed on a system, then installation of NumPy is very easy.

```
C:\Users\Your Name>pip install numpy
```

### Import numPy

Once NumPy is installed, import it in your applications by adding the import keyword.

```
import numpy
```

**Simple example**

```
import numpy
arr = numpy.array([1, 2, 3, 4, 5])
print(arr)
```

**Output:** [1 2 3 4 5]

**Creating Alias**

Alias is an alternative name for referring the same thing. NumPy is usually imported under the np aliases. Use as here: `import numpy as np`. Now the numpy package can be referred to as np instead of numpy.

See the difference

- `import numpy`  
`arr = numpy.array([1, 2, 3, 4, 5])`  
`print(arr)`
- `import numpy as np`  
`arr = np.array([1, 2, 3, 4, 5])`  
`print(arr)`

**ndarray data structure**

The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

**Creating arrays**

NumPy is used to work with arrays. The array object in NumPy is called ndarray. We can create a NumPy ndarray object by using `array()` function.

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
print(arr)
```

**Output:** [1 2 3 4 5]

To create an ndarray, we can pass a list, tuple or any array-like object into the `array()` method and it will be converted into an ndarray.

- Use a tuple to create a NumPy array:

```
import numpy as np
arr = np.array((1, 2, 3, 4, 5))
print(arr)
```

**Output:** [1 2 3 4 5]

**Dimensions in Arrays**

A dimension in arrays is one level of array depth. The nested arrays are the arrays that have arrays as their elements.

**0-D arrays**

0-D arrays, or Scalars, are the elements in an array. Each value in an array is a 0-D array. Example: Create a 0-D array with element 42.

```
import numpy as np
arr = np.array(42)
```

```
print(arr)
```

Output: 42

### 1-D arrays

An array that has 0-D arrays as its elements is called uni-dimensional or 1-D array. These are the most common and basic arrays. Example: Create a 1-D array containing the values 1,2,3,4,5:

```
import numpy as np
```

```
arr = np.array([1, 2, 3, 4, 5])
```

```
print(arr)
```

Output: [1 2 3 4 5]

### 2-D arrays

An array that has 1-D arrays as its elements is called a 2-D array. These are often used to represent matrix or 2nd order tensors. Example: Create a 2-D array containing two arrays with the values 1,2,3 and 4,5,6:

```
import numpy as np
```

```
arr = np.array([[1, 2, 3], [4, 5, 6]])
```

```
print(arr)
```

Output: [[1 2 3]

[4 5 6]]

### 3-D arrays

An array that has 2-D arrays (matrices) as its elements is called 3-D array. These are often used to represent a 3rd order tensor. Example: Create a 3-D array with two 2-D arrays, both containing two arrays with the values 1,2,3 and 4,5,6:

```
import numpy as np
```

```
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
```

```
print(arr)
```

Output: [[[1 2 3]

[4 5 6]]

[[1 2 3]

[4 5 6]]]

### Check the number of dimensions

NumPy Arrays provides the `ndim` attribute that returns an integer that tells us how many dimensions the array has. Example: Check how many dimensions the arrays have:

```
import numpy as np
```

```
a = np.array(42)
```

```
b = np.array([1, 2, 3, 4, 5])
```

```
c = np.array([[1, 2, 3], [4, 5, 6]])
```

```
d = np.array([[[1, 2, 3], [4, 5, 6]], [[1, 2, 3], [4, 5, 6]]])
```

```
print(a.ndim)
```

```
print(b.ndim)
```

```
print(c.ndim)
```

```
print(d.ndim)
```

Output:

0

1  
2  
3

### Higher Dimensional Arrays

An array can have any number of dimensions. When the array is created, you can define the number of dimensions by using the `ndmin` argument. Example: Create an array with 5 dimensions and verify that it has 5 dimensions:

```
import numpy as np
arr = np.array([1, 2, 3, 4], ndmin=5)
print(arr)
print('number of dimensions :', arr.ndim)
```

Output: [[[[[1 2 3 4]]]]]

Number of dimensions : 5

## 14.11 Operations on NumPy Arrays

### indexing

Array indexing is the same as accessing an array element. You can access an array element by referring to its index number. The indexes in NumPy arrays start with 0, meaning that the first element has index 0, and the second has index 1 etc.

- Example: Get the first element from the following array:

```
import numpy as np
arr = np.array([1, 2, 3, 4])
print(arr[0])
```

Output: 1

- Example: Get the second element from the following array:

```
import numpy as np
arr = np.array([1, 2, 3, 4])
print(arr[1])
```

Output: 2

- Example: Get third and fourth elements from the following array and add them.

```
import numpy as np
arr = np.array([1, 2, 3, 4])
print(arr[2] + arr[3])
```

Output: 7

Access 2-D arrays:

To access elements from 2-D arrays we can use comma separated integers representing the dimension and the index of the element. Think of 2-D arrays like a table with rows and columns, where the row represents the dimension, and the index represents the column.

- Example: Access the element on the first row, second column:

```
import numpy as np
arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print ('2nd element on 1st row: ', arr[0, 1])
```

Output: 2<sup>nd</sup> element on 1<sup>st</sup> row: 2

- Example: Access the element on the 2nd row, 5th column:

```
import numpy as np
arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
```

```
print ('5th element on 2nd row: ', arr[1, 4])
```

Output: 5<sup>th</sup> element on 2<sup>nd</sup> row: 10

Access 3-D arrays:

To access elements from 3-D arrays we can use comma separated integers representing the dimensions and the index of the element. Example: Access the third element of the second array of the first array:

```
import numpy as np
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
print (arr[0, 1, 2])
```

Output: 6

## Slicing

Slicing in python means taking elements from one given index to another given index. We pass slice instead of index like this: [start:end]. We can also define the step, like this: [start:end:step]. If we don't pass start its considered 0, if we don't pass end its considered length of array in that dimension and if we don't pass step its considered 1. Example: Slice elements from index 1 to index 5 from the following array:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[1:5])
```

Output: [2 3 4 5]

- Example: Slice elements from index 4 to the end of the array:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[4:])
```

Output: [5 6 7]

- Example: Slice elements from index 4 to the end of the array:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[:4])
```

Output: [1 2 3 4]

Negative slicing: Use the minus operator to refer to an index from the end.

- Example: Slice from the index 3 from the end to index 1 from the end:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[-3:-1])
```

Output: [5 6]

- Example: Return every other element from index 1 to index 5:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[1:5:2])
Output: [2 4]
```

- Example: Return every other element from the entire array:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6, 7])
print(arr[::2])
Output: [1 3 5 7]
```

Slicing 2-D arrays

Example: From the second element, slice elements from index 1 to index 4 (not included):

```
import numpy as np
arr = np.array([[1,2,3,4,5],[6,7,8,9,10]])
print (arr[1, 1:4])
Output: [7 8 9]
```

## 14.12 Data Types in NumPy

NumPy has some extra data types. It refers to data types with only one character, like I for integers, u for unsigned integers, etc. These are:

- i-integer
- b-boolean
- u-unsigned integer
- f-float
- c-complex float
- m-timedelta
- M-datetime
- O-object
- S-string
- U-Unicode string
- V-fixed chunk of memory for other type

### Checking the data type of an array

The numPy array object has a property called dtype that return the data type of the array.

```
import numpy as np
arr = np.array([1, 2, 3, 4])
print(arr.dtype)

import numpy as np
arr = np.array(['apple', 'banana', 'cherry'])
print(arr.dtype)
```

### 14.13 Creating Arrays

We use the `array()` function to create arrays. The `dtype()` function is used to define the data type of the array elements. For example: Create an array with data type string:

```
import numpy as np
arr = np.array([1, 2, 3, 4], dtype='S')
print(arr)
print(arr.dtype)
```

For `i`, `u`, `f`, `S` and `U` we can define the size as well. For example: Create an array with data type 4 bytes' integer:

```
import numpy as np
arr = np.array([1, 2, 3, 4], dtype='i4')
print(arr)
print(arr.dtype)
```

If a type is given in which elements can't be casted then NumPy will raise a `ValueError`. `ValueError`: In Python `ValueError` is raised when the type of passed argument to a function is unexpected/incorrect. Example: A non-integer string like 'a' cannot be converted to integer (will raise an error):

```
import numpy as np
arr = np.array(['a', '2', '3'], dtype='i')
```

#### **Converting data types on existing arrays**

The best way to change the data type of an existing array, is to make a copy of the array with the `astype()` method. The `astype()` function creates a copy of the array and allows you to specify the data type as a parameter. The data type can be specified using a string, like 'f' for float, 'i' for integer etc. or you can use the data type directly like float for float and int for int. For example: Change data type from float to integer by using 'i' as parameter value:

```
import numpy as np
arr = np.array([1.1, 2.1, 3.1])
newarr = arr.astype('i')
print(newarr)
print(newarr.dtype)
```

Example: Change data type from float to integer by using `int` as parameter value:

```
import numpy as np
arr = np.array([1.1, 2.1, 3.1])
newarr = arr.astype(int)
print(newarr)
print(newarr.dtype)
```

Example: Change data type from integer to boolean:

```
import numpy as np
arr = np.array([1, 0, 3])
newarr = arr.astype(bool)
print(newarr)
print(newarr.dtype)
```

## 14.14 NumPy Operations

### NumPy copy vs View

The main difference between a copy and a view of an array is that the copy is a new array, and the view is just a view of the original array. The copy owns the data and any changes made to the copy will not affect original array, and any changes made to the original array will not affect the copy. The view does not own the data and any changes made to the view will affect the original array, and any changes made to the original array will affect the view.

Example: Make a copy, change the original array, and display both arrays:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
x = arr.copy()
arr[0] = 42
print(arr)
print(x)
```

Example: Make a view, change the original array, and display both arrays:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
x = arr.view()
arr[0] = 42
print(arr)
print(x)
```

### Make changes in view

Example: Make a view, change the view, and display both arrays:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
x = arr.view()
x[0] = 31
print(arr)
print(x)
```

### Check if Array owns its data

Copies owns the data, and views does not own the data, but how can we check this? Every NumPy array has the attribute `base` that returns `None` if the array owns the data. Otherwise, the `base` attribute refers to the original object.

Example: Print the value of the `base` attribute to check if an array owns its data or not:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5])
x = arr.copy()
y = arr.view()
print(x.base)
print(y.base)
```



### 14.15 NumPy Array Shape

The shape of an array is the number of elements in each dimension. NumPy arrays have an attribute called shape that returns a tuple with each index having the number of corresponding elements. Example: Print the shape of a 2D array

```
import numpy as np
arr = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])
print(arr.shape)
```

Example: Create an array with 5 dimensions using ndim using a vector with values 1,2,3,4 and verify that last dimension has value 4:

```
import numpy as np
arr = np.array([1, 2, 3, 4], ndmin=5)
print(arr)
print ('shape of array :', arr.shape)
```

### 14.16 Reshaping NumPy arrays

Reshaping means changing the shape of an array. The shape of an array is the number of elements in each dimension. By reshaping we can add or remove dimensions or change number of elements in each dimension.

#### **Reshape from 1-D to 2-D**

Example: Convert the following 1-D array with 12 elements into a 2-D array. The outermost dimension will have 4 arrays, each with 3 elements:

```
import numpy as np
arr = np.array([1,2,3,4,5,6,7,8,9,10,11,12])
newarr = arr.reshape(4, 3)
print(newarr)
```

#### **Reshape from 1-D to 3-D**

Example: Convert the following 1-D array with 12 elements into a 3-D array. The outermost dimension will have 2 arrays that contains 3 arrays, each with 2 elements:

```
import numpy as np
arr = np.array([1,2,3,4,5,6,7,8,9,10,11,12])
newarr = arr.reshape(2, 3, 2)
print(newarr)
```

### 14.17 NumPy Array Iterating

Iterating means going through elements one by one. As we deal with multi-dimensional arrays in numpy, we can do this using basic for loop of python. If we iterate on a 1-D array it will go through each element one by one.

Example: Iterate on the elements of the following 1-D array:

```
import numpy as np
arr = np.array([1, 2, 3])
for x in arr:
 print(x)
```

Example: Iterate on the elements of the following 2-D array:

```
import numpy as np
arr = np.array([[1, 2, 3], [4, 5, 6]])
for x in arr:
 print(x)
```

Example: Iterate on the elements of the following 3-D array:

```
import numpy as np
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
for x in arr:
 print(x)
```

## 14.18 Joining NumPy Arrays

Joining means putting contents of two or more arrays in a single array. In SQL we join tables based on a key, whereas in NumPy we join arrays by axes. We pass a sequence of arrays that we want to join to the concatenate () function, along with the axis. If the axis is not explicitly passed, it is taken as 0.

- Example: Join two arrays

```
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.concatenate((arr1, arr2))
print(arr)
```

- Example: Join two 2-D arrays along rows (axis=1):

```
import numpy as np
arr1 = np.array([[1, 2], [3, 4]])
arr2 = np.array([[5, 6], [7, 8]])
arr = np.concatenate((arr1, arr2), axis=1)
print(arr)
```

### Joining NumPy Arrays using Stack Functions

Stacking is same as concatenation; the only difference is that stacking is done along a new axis. We can concatenate two 1-D arrays along the second axis which would result in putting them one over the other, ie. stacking. We pass a sequence of arrays that we want to join to the stack () method along with the axis. If the axis is not explicitly passed it is taken as 0.

Example:

```
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.stack((arr1, arr2), axis=1)
print(arr)
```

1. Stacking Along Rows: NumPy provides a helper function hstack() to stack along rows.

```
import numpy as np
arr1 = np.array([1, 2, 3])
```

```
arr2 = np.array([4, 5, 6])
arr = np.hstack((arr1, arr2))
print(arr)
```

2. Stacking Along Columns: NumPy provides a helper function `vstack()` to stack along columns.

```
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.vstack((arr1, arr2))
print(arr)
```

3. Stacking Along Height (depth): NumPy provides a helper function `dstack()` to stack along height, which is the same as depth.

```
import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr = np.dstack((arr1, arr2))
print(arr)
```

### **14.19 NumPy Splitting Arrays**

Splitting is the reverse operation of Joining. Joining merges multiple arrays into one and Splitting breaks one array into multiple. We use `array_split()` for splitting arrays, we pass it the array we want to split and the number of splits.



**Example:** Split the array in 3 parts:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 6])
newarr = np.array_split(arr, 3)
print(newarr)
```

The return value of the `array_split()` method is an array containing each of the split as an array. If you split an array into 3 arrays, you can access them from the result just like any array element:

### **14.20 NumPy Array Search**

You can search an array for a certain value and return the indexes that get a match. To search an array, use the `where()` method.



**Example:** Find the indexes where the value is 4:

```
import numpy as np
arr = np.array([1, 2, 3, 4, 5, 4, 4])
x = np.where(arr == 4)
print(x)
```

There is a method called `searchsorted()` which performs a binary search in the array and returns the index where the specified value would be inserted to maintain the search order.

### 14.21 NumPy Sorting arrays

Sorting means putting elements in an ordered sequence. Ordered sequence is any sequence that has an order corresponding to elements, like numeric or alphabetical, ascending or descending. The NumPy ndarray object has a function called `sort()` that will sort a specified array.



**Example:** Sort the array

```
import numpy as np
arr = np.array([3, 2, 0, 1])
print(np.sort(arr))
```

O/P: [0 1 2 3]

`Sort()`: This method returns a copy of the array, leaving the original array unchanged. You can also sort arrays of strings, or any other data type. Example: Sort the array alphabetically:

```
import numpy as np
arr = np.array(['banana', 'cherry', 'apple'])
print(np.sort(arr))
```

O/P: ['apple', 'banana', 'cherry']



**Example:** Sort a boolean array:

```
import numpy as np
arr = np.array([True, False, True])
print(np.sort(arr))
```

O/P: [False True True]

If you use the `sort()` method on a 2-D array, both arrays will be sorted. Example: Sort a 2-D array:

```
import numpy as np
arr = np.array([[3, 2, 4], [5, 0, 1]])
print(np.sort(arr))
```

O/P: [[2 3 4]

[0 1 5]]

### 14.22 NumPy Filter Arrays

Getting some elements out of an existing array and creating a new array out of them is called filtering. In NumPy, you filter an array using a boolean index list. A boolean index list is a list of booleans corresponding to indexes in the array. If the value at an index is `True` that element is contained in the filter array, if the element at that index is `False` that element is excluded from the filter array. Example: Create an array from the elements on index 0 and 2:

```
import numpy as np
arr = np.array([41, 42, 43, 44])
x = [True, False, True, False]
newarr = arr[x]
print(newarr)
```

O/P: [41 43]

**Example:** Create a filter array that will return only values higher than 42:

```
import numpy as np
arr = np.array([41, 42, 43, 44])
filter_arr = arr > 42
newarr = arr[filter_arr]
print(filter_arr)
print(newarr)
```

O/P: [43 44]

### 14.23 Random Number in NumPy

Random number does NOT mean a different number every time. Random means something that cannot be predicted logically. NumPy offers the random module to work with random numbers.

**Example:** Generate a random integer from 0 to 100:

```
from numpy import random
x = random.randint(100)
print(x)
```

The random module's rand method returns a random float between 0 and 1. Example: Generate a random float from 0 to 1:

```
from numpy import random
x = random.rand()
print(x)
```

### 14.24 Pandas

Pandas is one of the most famous data science tools and it is used for cleaning, manipulating, and analyzing data. The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008. Python has a powerful and most popular package 'Pandas' built on top of Numpy which has the implementation of many data objects and data operations.

### 14.25 Why Pandas?

- Pandas provide tools for reading and writing data into data structures and files.
- It also provides powerful aggregation functions to manipulate data.
- Pandas provide extended data structures to hold different types of labeled and relational data. This makes python highly flexible and extremely useful for data cleaning and manipulation.
- Pandas is highly flexible and provides functions for performing operations like merging, reshaping, joining, and concatenating data.
- Pandas allows us to analyze big data and make conclusions based on statistical theories. Pandas can clean messy data sets and make them readable and relevant. Relevant data is very important in data science.

## 14.26 Installing and Importing Pandas

File → Settings → python interpreter → + → search pandas → install package.

Once Pandas is installed, import it in your applications by adding the import keyword: `import pandas`. Pandas is usually imported under the `pd` alias. Create an alias with the `as` keyword while importing:

```
import pandas as pd
```

Now the Pandas package can be referred to as `pd` instead of `pandas`.

## 14.27 Data Structures of Pandas

There are two main data structures of Pandas. These are:

1. Series
2. Dataframes

### 1. Pandas Series

The Pandas Series is like a column in a table. It is a one-dimensional array holding data of any type. Example: Create a simple Pandas Series from a list:

```
import pandas as pd
```

```
a = [1, 7, 2]
```

```
myvar
```

```
= pd.Series(a)
```

```
print(myvar)
```

If nothing else is specified, the values are labeled with their index number. The first value has index 0, second value has index 1 etc. This label can be used to access a specified value. Example: Return the first value of the Series:

```
print(myvar[0])
```

With index argument, you can name your own labels. Example: Create your own labels.

```
import pandas as pd
```

```
a = [1, 7, 2]
```

```
myvar = pd.Series(a, index = ["x", "y", "z"])
```

```
print(myvar)
```

When you have created labels, you can access an item by referring to the label. Example: Return the value of "y":

```
print(myvar["y"])
```

Pandas Dataframes

Data sets in Pandas are usually multi-dimensional tables, called DataFrames. A Pandas DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns. Series is like a column, a DataFrame is the whole table.



**Example:** Create a DataFrame from two Series:

```
import pandas as pd
```

```
data={"calories":[420,380,390],"duration":[50,40,45]}
```

```
myvar = pd.DataFrame(data)
```

```
print(myvar)
```



**Example:** Create a simple Pandas DataFrame:

```
import pandas as pd
data={"calories":[420,380,390],"duration":[50,40,45]}
#load data into a DataFrame object:
df = pd.DataFrame(data)
print(df)
```

Pandas can read CSV files

A simple way to store big data sets is to use CSV files (comma separated files). CSV files contains plain text and is a well know format that can be read by everyone including Pandas. Example: Load the CSV into a DataFrame:

```
import pandas as pd
df = pd.read_csv('data.csv')
print(df.to_string())
```

Use `to_string` to print the entire DataFrame.

## 14.28 Data Cleaning

Data cleaning means fixing bad data in your data set. The bad data could be:

1. Empty cells
2. Data in wrong format
3. Wrong data
4. Duplicates

There are various ways to deal with these different things. These are:

### 1. **Cleaning Empty Cells**

- Empty cells can potentially give you a wrong result when you analyze data.
  - A. Remove rows: One way to deal with empty cells is to remove rows that contain empty cells. This is usually OK, since data sets can be very big, and removing a few rows will not have a big impact on the result. Example: Return a new Data Frame with no empty cells:

```
import pandas as pd
df = pd.read_csv('data.csv')
new_df = df.dropna()
print(new_df.to_string())
```

The `dropna()` method returns a new DataFrame and will not change the original. If you want to change the original DataFrame, use the `inplace = True` argument.

- B. Replace empty cells: Another way of dealing with empty cells is to insert a new value instead. This way you do not have to delete entire rows just because of some empty cells. The `fillna()` method allows us to replace empty cells with a value:
- C. Replace only for specified columns: To only replace empty values for one column, specify the column name for the DataFrame. Example: Replace NULL values in the "Calories" columns with the number 130:

```
import pandas as pd
df = pd.read_csv('data.csv')
df["Calories"].fillna(130, inplace = True)
```

- D. Replace using mean, median or mode: A common way to replace empty cells, is to calculate the mean, median or mode value of the column. Pandas uses the `mean()`, `median()` and `mode()` methods to calculate the respective values for a specified column:
- E. Cleaning data of wrong format

Cells with data of wrong format can make it difficult, or even impossible, to analyze data.

How to deal?

1. Convert into a correct format: Try to convert all cells in the 'Date' column into dates. Pandas has a `to_datetime()` method for this.
2. **Remove the rows:** We can remove the rows by using `dropna()` method. Example: Remove rows with a NULL value in the "Date" column: `df.dropna(subset=['Date'], inplace = True)`

### 3. Cleaning Wrong data

- "Wrong data" does not have to be "empty cells" or "wrong format", it can just be wrong, like if someone registered "199" instead of "1.99". Sometimes you can spot wrong data by looking at the data set, because you have an expectation of what it should be.

How to deal?

1. Replacing values: One way to fix wrong values is to replace them with something else. Example: Set "Duration" = 45 in row 7:  
`df.loc[7, 'Duration'] = 45`
2. Removing rows: Another way of handling wrong data is to remove the rows that contains wrong data. This way you do not have to find out what to replace them with, and there is a good chance you do not need them to do your analyses.

Duplicating Rows: Duplicate rows are rows that have been registered more than one time. To discover duplicates, we can use the `duplicated()` method. This method returns a Boolean value for each row. Duplicate rows are rows that have been registered more than one time.

Example: Returns True for every row that is a duplicate, otherwise False.

```
print(df.duplicated())
```

How to deal?

1) Removing duplicates: To remove duplicates, use the `drop_duplicates()` method. Example: Remove all duplicates:

```
df.drop_duplicates(inplace = True)
```

## 14.29 Data Transformation Operations

1. Rescaling data
2. Normalizing data
3. Binarizing data
4. Standardizing data
5. Label encoding
6. One hot encoding

### Summary

- Indentation refers to the spaces at the beginning of a code line. Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important. Python uses indentation to indicate a block of code.



- Comments can be used to explain Python code. Comments can be used to make the code more readable. Comments can be used to prevent execution when testing code. Comments can be created using a specific way in Python. Comments start with # and Python will ignore them.
- Strings are used to record textual information as well as arbitrary collections of bytes. A sequence in Python—that is, a positionally ordered collection of other objects. Sequences maintain a left-to-right order among the items they contain: their items are stored and fetched by their relative position.
- Python also comes with Booleans (with predefined True and False objects that are essentially just the integers 1 and 0 with custom display logic), and it has long supported a special placeholder object called None commonly used to initialize names and objects:
- The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

### Keywords

- **Interpreted:** An interpreted language is a programming language which are generally interpreted, without compiling a program into machine instructions. It is one where the instructions are not directly executed by the target machine, but instead read and executed by some other program.
- **Object-oriented:** Object-oriented language is a high-level computer programming language that implements objects and their associated procedures within the programming context to create software programs. Object-oriented language uses an object-oriented programming technique that binds related data and functions into an object and encourages reuse of these objects within the same and other programs.
- **High-level:** Python is a high-level language which just means that it's simpler for a human to use. Low-level languages such as C/C++ require a much more detailed understanding of how a computer works. With a high-level language, many of these details are abstracted away to make your life easier. Python uses dynamic semantics, meaning that its variables are dynamic objects.
- **Python variables:** Variables are containers for storing data values. Python has no command for declaring a variable. A variable is created the moment you first assign a value to it.
- **Sets:** These are neither mappings nor sequences; rather, they are unordered collections of unique and immutable objects. Sets are created by calling the built-in set function or using new set literals and expressions in 3.0, and they support the usual mathematical set operations (the choice of new {...} syntax for set literals in 3.0 makes sense, since sets are much like the keys of a valueless dictionary).

### SelfAssessment

1. Python is
  - A. High level language
  - B. Interpreted
  - C. Object oriented
  - D. All of the above
2. Which of the following are the features of python programming language?

- A. Open source
  - B. Powerful
  - C. Simple
  - D. All of the above
3. On which platform, Python language can work?
- A. Windows
  - B. Linux
  - C. Mac
  - D. All of the above
4. Which symbol is used to specify a comment in Python?
- A. #
  - B. \$
  - C. %
  - D. ^
5. The variable name `my_variable_name` represents which case?
- A. Pascal case
  - B. Snake case
  - C. Camel case
  - D. None of the above
6. Which of the following are data transformation operations?
- A. Normalizing data
  - B. Binarizing data
  - C. Standardizing data
  - D. All of the above
7. Which of the following are immutable in nature?
- A. Strings
  - B. Tuples
  - C. Both Strings and tuples
  - D. None of the above
8. Python language is used in
- A. Web applications
  - B. Desktop applications
  - C. Cybersecurity
  - D. All of the above
9. Which of the following are the data structures of Pandas?
- A. Series
  - B. Dataframes
  - C. Both of the above
  - D. None of the above

10. Which of the following is a multi-dimensional table?
  - A. Series
  - B. Dataframes
  - C. Both of the above
  - D. None of the above
  
11. The hstack() function is for
  - A. Stacking along rows
  - B. Stacking along columns
  - C. Stacking along height
  - D. None of the above
  
12. The strings data types of python supports
  - A. Indexing
  - B. Slicing
  - C. Concatenation
  - D. All of the above
  
13. Which of the following is the mutable data type in Python?
  - A. List
  - B. Tuple
  - C. String
  - D. None of the above
  
14. The lists are
  - A. Positionally ordered
  - B. No fixed size
  - C. Mutable
  - D. All of the above
  
15. Which of the following does not own the data in numPy?
  - A. Copy
  - B. View
  - C. Both of the above
  - D. None of the above

### Answers for Self Assessment

- |       |       |       |       |       |
|-------|-------|-------|-------|-------|
| 1. D  | 2. D  | 3. D  | 4. A  | 5. B  |
| 6. D  | 7. C  | 8. D  | 9. C  | 10. B |
| 11. A | 12. D | 13. A | 14. D | 15. B |

### Review Questions

1. What is Python? Write about its features in detail.

2. How can we compare Python with other programming languages? Write the differences.
3. What is numPy? What kind of operations can be performed on it?
4. What is Pandas? What are the different data structures it has?
5. What is data cleaning? Which different strategies are used for cleaning the data?

**Further Readings**

<https://www.ibm.com/in-en/cloud/learn/machine-learning>

**Web Links**

<https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML>

**LOVELY PROFESSIONAL UNIVERSITY**

Jalandhar-Delhi G.T. Road (NH-1)  
Phagwara, Punjab (India)-144411  
For Enquiry: +91-1824-521360  
Fax.: +91-1824-506111  
Email: [odl@lpu.co.in](mailto:odl@lpu.co.in)

