DATA SCIENCE

CLASS XI

Version 1.0

Microsoft

DATA SCIENCE

GRADE XI

Student Handbook

ACKNOWLEDGMENT

Patrons

- Sh. Ramesh Pokhriyal 'Nishank', Minister of Human Resource Development, Government of India
- Sh. Dhotre Sanjay Shamrao, Minister of State for Human Resource Development, Government of India
- Ms. Anita Karwal, IAS, Secretary, Department of School Education and Literacy, Ministry Human Resource Development, Government of India Advisory

Editorial and Creative Inputs

- Mr. Manuj Ahuja, IAS, Chairperson, Central Board of Secondary Education

Guidance and Support

- Dr. Biswajit Saha, Director (Skill Education & Training), Central Board of Secondary Education
- Dr. Joseph Emmanuel, Director (Academics), Central Board of Secondary Education
- Sh. Navtez Bal, Executive Director, Public Sector, Microsoft Corporation India Pvt. Ltd.
- Sh. Omjiwan Gupta, Director Education, Microsoft Corporation India Pvt. Ltd
- Dr. Vinnie Jauhari, Director Education Advocacy, Microsoft Corporation India Pvt. Ltd.
- Ms. Navdeep Kaur Kular, Education Program Manager, Allegis Services India

Value adder, Curator and Co-Ordinator

- Sh. Ravinder Pal Singh, Joint Secretary, Department of Skill Education, Central Board of Secondary Education

ABOUT THE HANDBOOK

In today's world, we have a surplus of data, and the demand for learning data science has never been greater. The students need to be provided a solid foundation on data science and technology for them to be industry ready.

The objective of this curriculum is to lay the foundation for Data Science, understanding how data is collected, analyzed and, how it can be used in solving problems and making decisions. It will also cover ethical issues with data including data governance and builds foundation for AI based applications of data science.

Therefore, CBSE is introducing 'Data Science' as a skill module of 12 hours duration in class VIII and as a skill subject in classes IX-XII.

CBSE acknowledges the initiative by Microsoft India in developing this data science handbook for class XI students. This handbook introduces to R programming in Data Science, solving practical examples. The course covers the theoretical concepts of data science followed by practical examples to develop critical thinking capabilities among students.

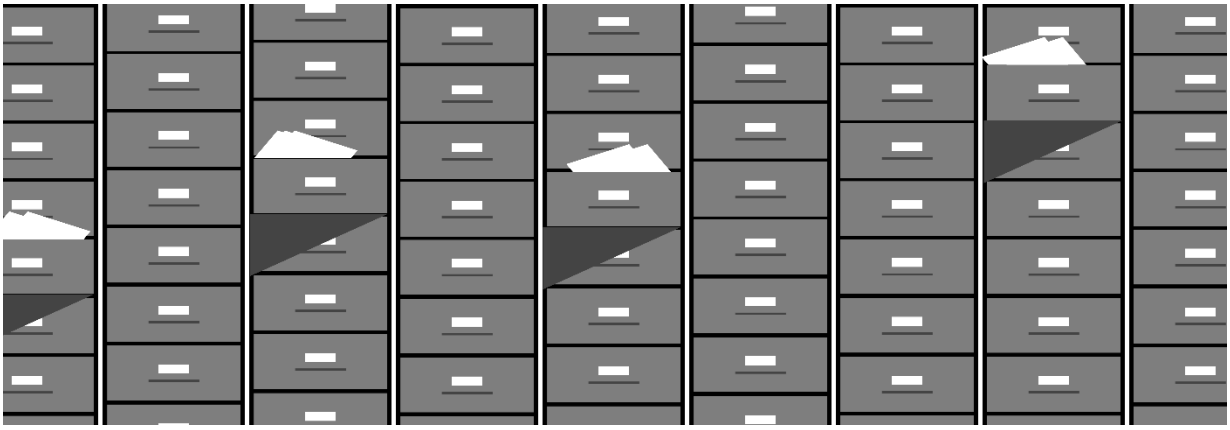
The purpose of the book is to enable the future workforce to acquire data science skills early in their educational phase and build a solid foundation to be industry ready.

Contents

CHAPTER

1

Ethics in Data Science



Studying this chapter should enable you to understand:

- Evolution of Data Ecosystem
- Need for Data scientists to understand ethics
- Concept of Data governance framework and its benefits.

1. Introduction

In this chapter, we will learn about ethical guidelines and data governance frameworks in data science.

2. How Data Ecosystem is evolving

In the initial stages of data science, the kind of data we all dealt with, be it for academic purposes or business needs was small, structural, and static. This is the kind of data that was easy to put into rows and columns and displayed via spreadsheets. In short, we can say that this was a happy place to be in for statisticians. Here, the traditional tools such as descriptive statistics, predictive modeling, and classifications were used to serve the purpose.

However, as data continued to evolve, it did not remain small, structural, and basic. The data that evolved became large, unstructured, and in motion. This change in data behavior led to the need to develop skills that look different from the skills required to look at data when the data was small, structural, and basic. Here, one must learn about sensor-based data, IoT data, machine

learning skills, and concepts like support vector machines. In short, the things that one needed to know in this scenario, to translate large, unstructured data into information, were different from those needed to know when data was small, structured, and basic.

The kind of data that we see today is massive, integrated, and dynamic. Here we are talking about a system of data talking to another system of data. To execute such behavior, one must learn concepts of machine learning and deep networks.

The figure shown below highlights the evolution of the data ecosystem.

Here, one must remember that the issues related to data ethics when data is small, structured, and basic are very simplistic. However, the issues get complicated when questions need to be asked to retrieve information from massive, integrated, and dynamic data.

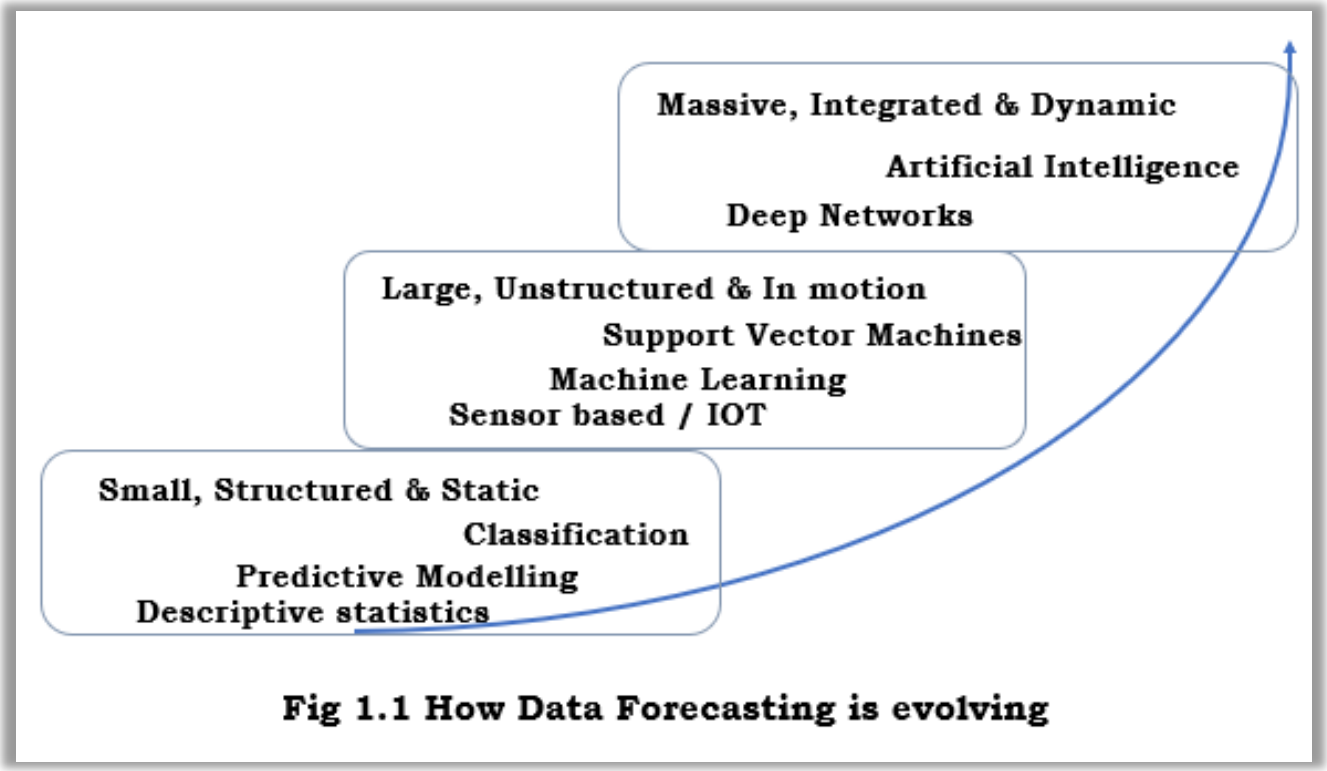


Fig 1.1 How Data Forecasting is evolving

3. Why do Data Scientists need to understand ethics?

Since data scientists have access to a vast pool of data in their data analysis, it becomes essential for them to adhere to ethical guidelines.

The use of protective mechanisms and policies to discourage the mishandling and unethical use of data should be made part of best practices.

Some of the negative scenarios that may arise if ethical guidelines are disrespected include:

a) A few people can do an immense amount of harm:

In the last decade, we have seen many organizations becoming vulnerable to data breaches. Hackers worldwide are on the lookout to crack through a reputed organization's firewalls and steal important data from their servers. The stolen data are then sold out for a hefty sum.

To date, Yahoo holds the title for the largest data breach in the history of the internet. In 2016, the company disclosed that it had been the victim of multiple data breaches over the years, starting in 2013. The data breaches had exposed the email addresses, names, dates of birth of around three billion people who have used Yahoo.

Another example of a data breach is Marriott (Starwood) hotel. In 2018, Marriott's data team had confirmed that around 383 million accounts of the guests were compromised in the year 2016. The breach had exposed the names, addresses, contact numbers, and passport information of the guests whose accounts were hacked.

b) Lack of consent:

One of the leading social networking sites experimented wherein without consent they purposely fed the users in their newsfeed highly extreme point of view, particularly incendiary part of the news feed because they were trying to elicit a reaction from the users and then to see if that impacted what the users were posting back. The newsfeed was curated with the purposeful intention to see if that ultimately impacted the way a user interacted with the rest of the network. Was the consent from the users taken before performing this experiment? The answer to this is "No".

4. What is data governance framework?

Data governance framework can be defined as a collection of practices and processes that ensure the authorized management of data in an organization. The primary purpose behind implementing data governance by any organization is to achieve better control over its data assets, including the methods, technologies, and behaviors around the proper management of data. It also deals with the security, privacy, integrity, and management of data flowing in and out of the organization.

Some of the benefits of implementing a data governance framework are:

1. Procedures around regulation and compliance activities become exact.
2. There is greater transparency within data-related activities.
3. Increase in value of organization's data.
4. Better resolution of issues around current data.

Recap

- In this chapter we have learnt that data over time has evolved both in size as well as in complexity.
- With increase in the volume of data getting generated every day, data scientists gathering data to perform analysis need to understand the importance of ethics.
- We have also learnt about data governance framework and its benefits

Exercises

Objective Type Questions

1) A person runs a small business and keeps all his/her business records on an unprotected personal computer. These records include essential information about his/her customers. Since it is a small business, that person believes that he/she is unlikely to be a target for hackers. According to him/her, several years have passed, and information on his/her unprotected computer has never been compromised. Are the actions of that person ethical?
   a) Yes
   b) No

2) One comes up with an idea to improve the way patient data is collected into electronic medical records, thereby reducing errors and better integrating data entry with patient care workflow. When an experiment is run to evaluate the idea, what kind of data is expected to be used?
   a) Prospective data
   b) Retrospective data

3) A supermarket has prominently displayed boards at various places in the store "We videotape you for your security". Later, you find out that the supermarket analyzes videos to decide store layout and product placement. You feel that the signage is misleading since the store uses the videos not just for security but also to boost profits. Are the supermarket's actions ethical?
   a) Yes
   b) No

4) Suppose a celebrity goes to a supermarket for shopping. The next day, images of the celebrity taken from the supermarket's video camera appear on a leading tabloid. Is the supermarket right in selling the images to the tabloid?
   a) Yes
   b) No

5) Once I have voluntarily shared some information about myself on the web, it means that this information is no longer private and can be shared freely.
   a) True
   b) False

6) Undesired analysis of previously collected personal data violates privacy.
   a) True

b) False

7) Undesired dissemination of previously collected data violates privacy.
   a) True
   b) False

Standard Questions

1) Explain in detail how data has evolved over time.
2) Explain with relevant examples why data scientists need to understand data and follow data ethics?
3) What is data governance framework?
4) What are some of the benefits of implementing data governance framework?

Higher Order Thinking Skills (HOTS)

Please answer the questions below in no less than 200 words.

1) What, according to you, should be the ethical principles for conducting research that involves dealing with other people's data?
2) Should there be differences in expectations about what is ethical online versus offline regarding handling of data?

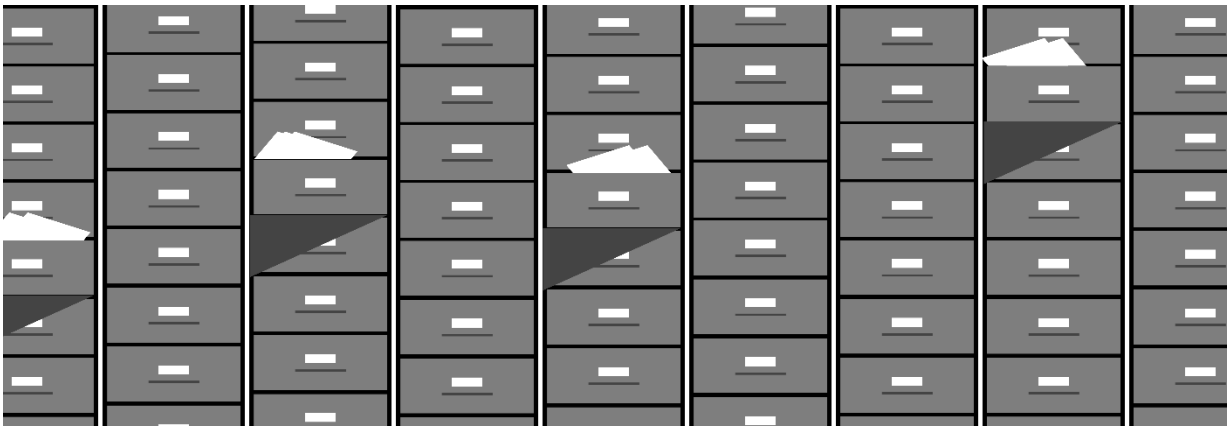
Applied Project

Suppose you have visited a restaurant for dining. At the end of the meal, the restaurant manager provides you with a form wherein you need to fill up your details along with your contact number. According to the manager, the purpose for collecting this data is to enable them to inform us of the exciting deals as and when they come up. Explain in detail, the precautions you need to take before handing out your details to the restaurant manager.

CHAPTER

2

Assessing Data



Studying this chapter should enable you to understand:

- Difference between story and fact
- Trial assessment in detail

1. Introduction

In the previous chapter, we learnt about ethical guidelines and data governance framework in data science.

In this chapter, we will learn to make distinction between story and fact. We will also explore the various aspects involved in performing trial assessment and the insights these assessments generate.

2. Story vs. facts

A story is an account of experiences of events presented by someone. A story may contain disproportionate weightage either in favor of or against an idea or thing. You can have ten different people go through the same set of events, and each of them may present a completely different experience for those set of events.

On the other hand, a fact is something that has occurred or occurs. In other words, it is a truth that has either happened or continues to happen in this universe. In general, people generally inspect a fact or a series of facts to derive a conclusion.

3. Trial assessment

A trial assessment is a set of steps executed to support, reject or confirm an assumption.

Concept of correlation and causation

In statistics, two or more variables are related if their values alter so that the rise or fall in one variable's value is either directly or inversely proportional to the rise or fall in the other variable's value.

In statistics, correlation describes the direction of a relationship between two or more variables. However, we cannot assume that change in one variable gives rise to the other variables' change. E.g., an Increase in sales of winter care products in the United States of America is correlated to the increase in summer care products in Australia.

On the other side, causation shows that one events' occurrence originates from the other events' occurrence. E.g., How different human activities, livestock farming, rising emissions, and cutting down trees in forests ultimately affect temperature.

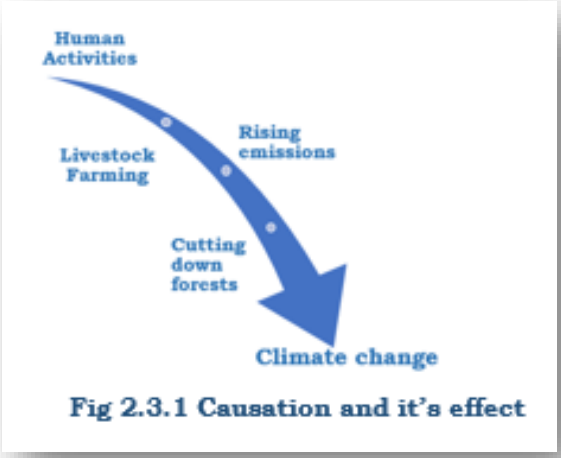


Fig 2.3.1 Causation and it's effect

Role of causation in a trial assessment

The idea behind performing the trial assessment is to test something. In other words, a trial assessment can be referred to as an experiment. There are usually two sets of variables in every experiment: the treatment variable and the response variable.

By treatment variable, we refer to the procedure variable. The treatment variable is generally an independent variable. On the other hand, the response variable is a dependent variable. In statistics, an experiment is defined as a supervised study in which a researcher tries to understand the cause and effect relationship.

Based on the analysis, the researcher concludes that the treatment followed had a causal effect on the response variable.

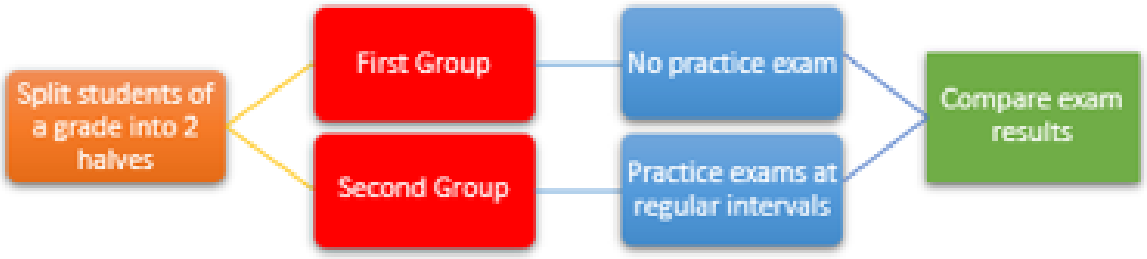


Fig 2.3.2 Example of cause and effect on students of a grade

Let us perform a trial assessment

One needs to perform a trial assessment to understand what is meant by the cause and effect relationship.

To start, let us take all the students of a grade in an academic year and split

them into two halves. The first half is subjected to the treatment of no practice exam. On the other hand, the second half is subjected to the treatment of practice exams regularly. At the end of the academic year, both the groups' annual exam results are compared. (Illustration is shown above)

Perception of time assessment

The perception of time assessment highlights a person's subjective experience of time duration within an ongoing event. This perceived duration can alter significantly between different individuals in different circumstances.

Imagine a person taking a launch chamber ride in a water park. In this ride, riders enter the chambers in nearly vertical positions. There is typically a countdown after which a trap door opens on the chamber floor to release the passenger. The anticipation and almost 90-degree launch make these among the most thrilling water park rides. In this case, let us consider that

Fig 2.3.3 Perception of time

the average duration of the ride is around 5 seconds.

Suppose we interview anyone who has just completed the ride for the first time and ask him/her how long that ride

lasted. In that case, that person will estimate around 10 – 11 seconds, which is more than the ride's actual duration.

4. Activity

The effects of different duration of light on the growth of radish seedlings?

A class of biology students in a school presented a question – What are the effects of duration of light on the growth of radish seedlings?

To answer the above question, the students designed and carried out an experiment using the following steps:

Collect/consider data

Data were required to be collected to answer the above question. The radish seeds were exposed to three different treatments – 24 hours of light (light), 12 hours of light and 12 hours of darkness (mixed), and 24 hours of darkness. The above three treatments covered two extreme cases and one in the middle.

With the assistance of a teacher, the students agreed to use plastic bags as a growth setup. The plastic bags allowed the students to observe and measure the seeds' germination without interfering with them. Two layers of moist paper towels were put inside a clear plastic bag. The paper towels were stapled in a line about one-third from the bag's bottom to hold the paper towel in place and provide a seam to hold the radish seeds.

For the study, one hundred twenty seeds were available. The growth setup allowed only thirty seeds. The class agreed to use the additional seeds and create a total of four growth setups – one for the light treatment, one for the mixed treatment, and two for the dark treatment. Thirty seeds were selected randomly and placed along the stapled seam of the light treatment bag. Next, thirty seeds were again chosen from the remaining ninety seeds and were placed inside the mixed treatment bag. Finally, thirty of the remaining sixty seeds were randomly chosen and placed inside one of the dark treatment bags. The last thirty seeds were placed inside the other dark treatment bag. Care was taken to make the four growth setups as identical as possible. With two growth setups for the same condition, their results could be compared to ensure similar handling. Three days later, the length of the radish seedlings for the germinating seeds was measured in millimeters.

The data were noted in a summary format like the one shown here.

1 - Light	**x, x, 2, 3, 5, 5, 5, 5, 5, 7, 7, 7, 8, 8, 8, 9, 10, 10, 10, 10, 10, 10, 10, 10, 14, 15, 15, 20, 21, 21**
2 - Mixed	**x, x, 3, 4, 5, 9, 10, 10, 10, 10, 10, 11, 13, 15, 15, 15, 17, 20, 20, 20, 20, 20, 21, 21, 22, 22, 23, 25, 25, 27**
3 - Dark	**x, 5, 8, 8, 10, 10, 14, 15, 15, 15, 20, 20, 20, 20, 20, 22, 25, 25, 25, 25, 26, 30, 30, 35, 35, 35, 35, 36, 37, 38**
3 - Dark	**x, 5, 8, 8, 10, 10, 10, 11, 14, 15, 15, 15, 16, 20, 20, 20, 20, 20, 24, 25, 29, 30, 30, 30, 30, 31, 33, 35, 35, 40**

Fig 2.4.1 Table showing Radish Seedling length after 3 days (sorted)

The table shows the sorted values. In each treatment type, some seeds did not germinate. Such values were considered missing values and were recorded as "x". Thus, there were 114 observations (28 for light treatment, 28 for mixed treatment, 58 for dark treatment).

It would have been more engaging if the students were encouraged to discuss whether excluding the seeds that did not germinate could add bias to the conclusions. In this scenario, because the number of seeds in each category was roughly the same, the missing values likely happened by chance. Conversely, if all the missing values were in one category, it would have suggested that that category's conditions hindered the growth, and so missing data were not accidental.

The data could be represented in another table with each observation (seed) on a separate row and each variable on a separate column for analysis purposes.

Seed #	Growth Bag	Treatment	Length(mm)
1	1	1-Light	x
2	1	1-Light	x
3	1	1-Light	2
.....	.....	.....	.....
118	4	3-Dark	35
119	3	3-Dark	35
120	4	3-Dark	40

Fig 2.4.2 Long Format Listing of Radish Seedling Lengths.

In the above table, the observed units are individual seeds. **Growth bag** indicates the bag (1, 2, 3, or 4) in which the seed is present, and **treatment** indicates the treatment (1-Light, 2-Mixed, or 3-Dark) the seeds are receiving. Both growth bag & treatment are said to be categorical variables. On the other hand, length is a quantitative variable, measuring the length of the seedlings in millimeters.

When the mean, median, and standard deviation were calculated on the seed samples exposed to different treatments (1-Light, 2-Mixed, or 3-Dark), the results produced were as shown in the table here:

Treatment	n	Mean	Median	StdDev.
1-Light	28	9.64	9.50	5.03
2-Mixed	28	15.82	16	6.76
3-Dark	28	21.86	20	9.75

Fig 2.4.3 Treatment Summary Statistics

Based on the results produced, the students might have come up with few questions:

a) Was there proof that 12 hours of light and 12 hours of dark group had a significantly higher mean length than the 24 hours of light group?
b) Was there proof that 24 hours of dark group had a significantly higher mean length than the 12 hours of light and 12 hours of dark group?
c) Were these differences in mean large enough to rule out their occurrences by chance as a possible clarification for the observed difference?

The mean length of the seedlings in the Treatment 2 – Mixed group was 6.2 mm more than the treatment 1 – Light group's mean length. Even though there was a difference of 6.2 mm, it might not be massive enough to rule out chance, and so it might become difficult to claim a treatment effect. This noticeable difference might have been due to one bag being fortunate enough to get a large count of good seeds. It could also be due to the light and water not being uniformly distributed among the treatment groups. But if a difference this large (6.2 mm) was probably the result of the randomization of seeds, then differences of such magnitude would have been observed quite often, if the measurements were rejumbled and a new difference in observed mean was calculated.

The students could have mocked this rejumbling by noting down each seedling's length on a separate card. Thus, for the 56 seedlings, there would have been 56 cards. These cards would then be shuffled and divided into two piles of 28 cards each. The first pile would represent the 1-Light treatment group and the second pile would represent the 2-Mixed treatment group. The students would have then computed the difference in mean between the two piles. The difference generated this way would have been entirely due to chance. By repeating the above process multiple times, the student would have able to interpret how the difference in mean varies when the treatment has no effect on the growth of the seedlings.

Figure 2.4.4 shown herewith was produced when technology was used to mix the growth measurements from 1-Light treatment and 2-Mixed treatment together and haphazardly divide the measurements into two groups of 28.

Here difference in mean was recorded, and the operation was repeated 200 times in total.

The observed difference of 6.2 mm was never exceeded in the simulation of 200 rejumbling. This gave strong proof against the supposition that the difference between means for treatments 1 and 2 was due to chance alone.

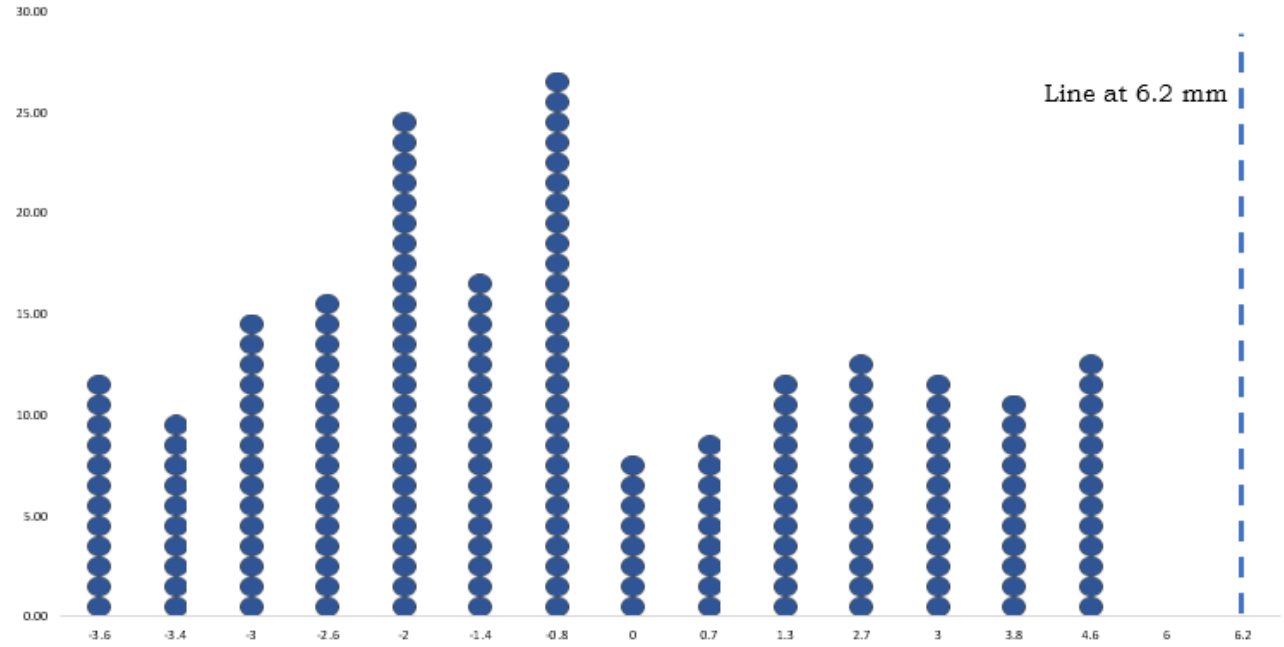


Fig 2.4.4 Difference in means of radish seedlings

When a similar type of procedure was followed for samples of treatment 2 and treatment 3, the observed difference of 6 mm in the mean length was never observed.

Figure 2.4.5 shown herewith was produced when a similar procedure as stated above was done with samples of treatment 2 and treatment 3.

Thus. here too, it gave a strong proof that the observed difference in mean length between treatment 2 and treatment 3 was not due to chance alone.

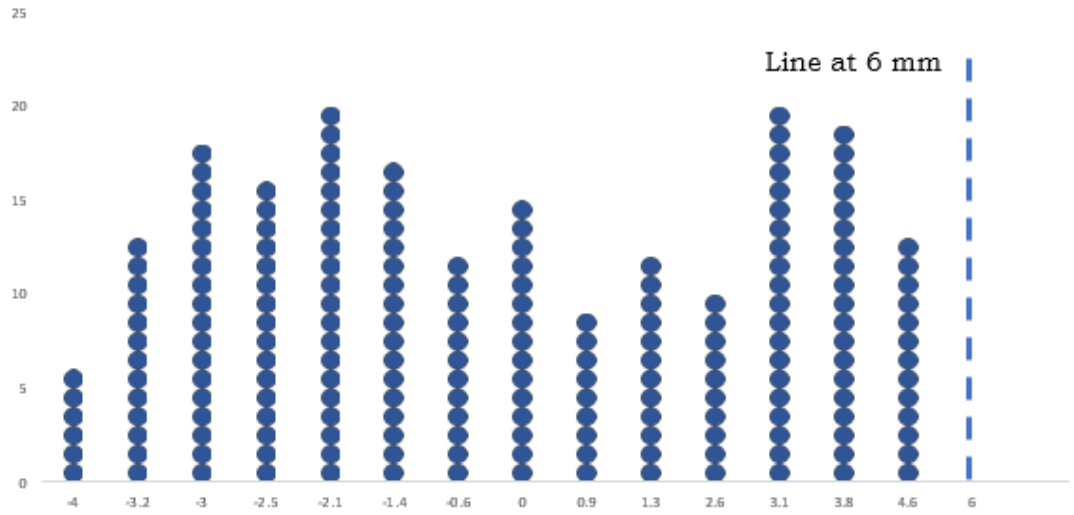


Fig 2.4.5 Difference in means of radish seedlings

Recap

- In this chapter we have learnt that there exists a difference between a story and fact.
- We have learnt about the concept of correlation and causation
- We have learnt how causation affects the outcome of a trial assessment
- We have learnt how a person perceives time under different circumstances

Exercises

Objective Type Questions

1) Which of the following is incorrect about a story?
   a) A story generally consists of one or more character(s)
   b) A story represents the point of view of a person
   c) A story has a theme associated with it
   d) Things revealed in a story are always correct and exist in the real world.

2) What is the nature of the correlation of two variables when they move in the same direction?
   a) Neutral
   b) Negative
   c) Positive
   d) None of the above

3)

Height (in cm.)	160	165	170	175	180	185
Weight (in Kg)	65.1	67.9	70.1	72.8	75.4	77.2

The correlation between height and weight in the above chart can be described as:
   a) Positive
   b) Negative
   c) Zero
   d) None of the above

4) In a study of insect life near a stream, data about the number of different insects' species and the distance from the stream were collected

Distance (in feet.)	2	5	8	11	14	17
Insect species	26	25	19	19	14	9

The correlation between the distance from the stream and the number of different species found is:

   a) Positive
   b) Negative
   c) Zero
   d) None of the above

5) Which of the following is an example of NO correlation?

a) The age of a child and his/her shoe size.
b) The age of a child and his/her height.
c) The age of a child and the number of pets owned.
d) The age of a child and vocabulary of words learned.

6) Which one of the below-mentioned scenarios appears most likely to be from causation?
   a) Reading one hour per day increases vocabulary.
   b) People who are homeless are more likely to have mental health issues.
   c) The weight of a person has nothing to do with the risk of heart disease.
   d) In India, as car sales increase, the birth rate also increases.

7) A study outlines that people who run more outdoors have higher rates of skin disease than people who exercise indoors. Which one of these seems the most likely to be connection between running and diseases?
   a) Running produces a chemical that causes skin disease.
   b) People who run generally drink more water and sports drinks, which might weaken the immune system's ability to attack disease germs.
   c) People who run do so because they are obese and might have poor health conditions, to begin with.
   d) People who run outdoors spend more time in the sun. Thus, they are exposed to harmful sunlight for more extended periods.

8) Which of the following statements given below shows a causal relationship and not just a correlated one?
   a) An individual's decision to work in construction and his/her diagnosis of skin disease.
   b) A decrease in temperature and an increase in the presence of people at ice skating rink.
   c) As the weight of a child increases so does his/her vocabulary.
   d) The time spent exercising and the number of calories burned.

9) For a person having a pleasant experience while doing something, time seems to:
   a) pass slowly
   b) come to a standstill
   c) fly quickly
   d) None of the above

Standard Questions

1) Write down three instances of stories and justify why you think they are stories?

2) Write down three facts and justify why you think they are facts?
3) What is correlation? (Support your answer with two examples of correlation)
4) What is the difference between positive and negative correlation?
5) What is causation? (Support your answer with two examples of causation)

Higher Order Thinking Skills (HOTS)

Please answer the questions below in no less than 200 words.

1) Is there a correlation between speaking and writing skills of an individual?

2) If outdoor runners have higher skin disease occurrences due to time exposure in the sun, think of an independent variable that can be used to test the relationship between running and skin diseases. What alternatives can one look for to negate the problem arising out of running outdoors in the sun?

3) Suppose you are reading a newly published fiction of your favorite author. The story turns out to be thoroughly engrossing, according to you. Describe your personal experience while reading the book about the time you took to complete the reading. (In the description, please mention whether it felt like it is taking too long to complete or was it the other way around). Once you have described your experience, kindly note down what you can infer from this.

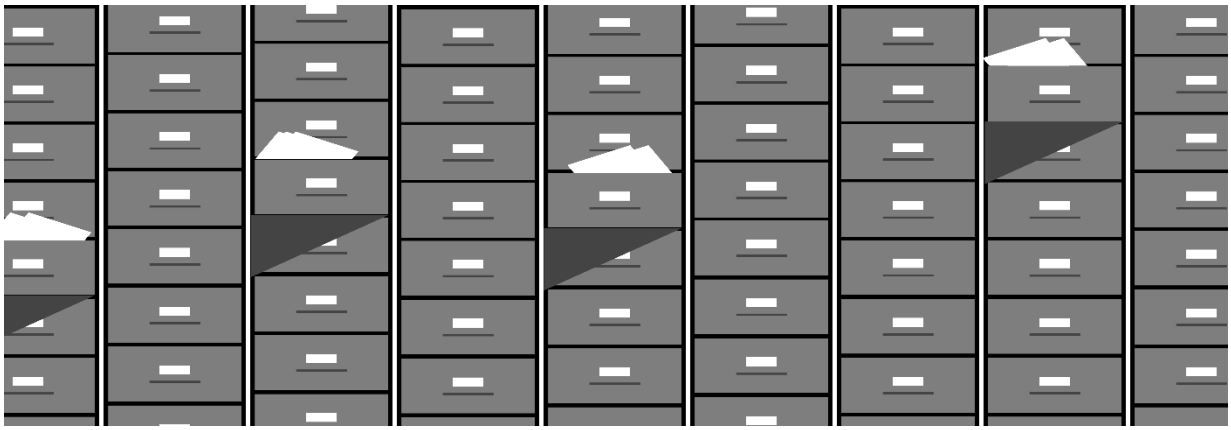
Applied Project

Consider a company manufacturing metal utensil for home cookware. The company has a factory in which 200 workers work on an everyday basis. Describe in detail how the set-up of the factory (proper equipment, safety measure and infrastructure) has a correlation and causal relationship with workers of the factory. Also, try to derive the correlation between workers of the factory, the profitability and sustenance of the company. One should also think about the end product the customers will be using and what difference it may cause to their lives.

CHAPTER

3

Forecasting on Data



Studying this chapter should enable you to understand:

- Forecasting
- Observational study
- Need for observational study
- Pros and cons of observational study

1. Introduction

In the previous chapter, we learned about the differences between story and fact. We also studied trial assessment in detail.

In this chapter, we are going to learn about forecasting and observational study.

2. Forecasting

Given all the information available, including the present and the historical data, forecasting can be defined as a statistical task that predicts the future as accurately as possible.

Fig 3.1 Forecasting

3. Observational study

An observational study can be defined as a procedure in which the subjects are just observed, and the results are then noted. During the investigation, nobody tries to interfere with the subject to affect the outcome.

Fig 3.2 Observational study

An example of an observational study would be if a researcher were trying to determine the outcome eating of organic diet has on overall health. The researcher finds 500 individuals, where 250 have eaten an organic diet in the past five years, while the rest 250 have not had an organic diet in the past five years. An overall health assessment is then performed on each of these 500 individuals. The result data from the health assessment are then analyzed, and conclusions are drawn on how an organic diet can affect one's overall health.

Why observational study when there is trial assessment?

Sometimes, it is not possible to perform trial assessments. In those scenarios, we need to rely upon observational study for data collection.

The reasons behind this are as follows:

1. In trial assessments, the subject is assigned to a random treatment and control group. However, it is unethical to expose the subject to arbitrary treatment in specific scenarios. Thus, observational studies are preferred over trial assessments.

   For example, purposefully exposing a subject to polluted air to observe the health issues that come to the forefront is unethical.

2. Large sums of money may be required to execute some of the trial assessments. There may be occasions when such large sums of money cannot be arranged. In such scenarios, it will be a better idea to drop the idea of performing trial assessments and give the observational study a priority.
3. A trial assessment cannot be performed in some scenarios as it becomes unfeasible to assign a subject to a group randomly.

Advantages of observational study

The advantages of observational study are as follows:

1. Observation is one of the simplest and most used methods of data gathering. Everybody in this world observes many things in their lives. With little training, one can become an expert in monitoring one's surroundings.

2. Another advantage of using an observational study is that since the observations are made in a perfectly natural setting, the analysis can reveal deep and unexplored insights. The revelation of such insights will be a rarity if we try to collect data via other means like surveys.

Disadvantages of observational study

The disadvantages of observational study are as follows:

1. Sometimes, the insights gained by an observational study are not justified by the amount of time spent to do so.

2. Certain events are uncertain and may not occur in the presence of an observer.

3. Sometimes, the observer may miss reporting important observational details.

4. The chances for unfair conclusion increase significantly in cases where an expert has not performed the study's analysis.

Recap

- In this chapter we have learnt about forecasting and observational study
- We have understood the need to perform observational study even though the option to perform trial assessment exists.
- We have learnt the advantages and disadvantages of observational study

Exercises

Objective Type Questions

Please choose the correct option in the questions below.

1. In forecasting, past and present data are used to predict the future as accurately as possible.
    a) The above statement is always true.
    b) The above statement is never true.
    c) The above statement is sometimes true.
    d) None of the above.

2. If the actual demand for a period is 100 units but forecast demand was 90 units. The error in forecast is
    a) -10
    b) +10
    c) -5
    d) +5

3. Observational study cannot be used in:
    a) Child studies
    b) Study about attitudes
    c) Animal studies
    d) Studies involving groups

4. Which of these is not true?
    a) Observational study is cheap.
    b) Observational study replaces interviewing.
    c) Observational study is time consuming.
    d) Observational study requires operational definition.

5. Which of these would make an observational study unethical?
    a) Putting an observer at risk of harm.
    b) Using multiple observers.
    c) Not getting consent from those being observed.
    d) Conducting the observation late at night.

6. Observer's reliability is improved by:
    a) Training observers.
    b) Using operational definitions

c) Restricting observations to specific time points
d) All the above

7. Which of the following is not a disadvantage of observational study?
   a) We need to assign the subject to random treatment and control groups.
   b) Certain events may occur in the absence of the observer.
   c) Miss the reporting of critical observational details.
   d) Time spent is far more compared to the insights gained via observation.

Standard Questions

1. What is forecasting? Give two examples of forecasting.
2. State the reasons why sometimes observational study is preferred over trial assessment?
3. What are the advantages of observational study?
4. What are the disadvantages of observational study?

Higher Order Thinking Skills (HOTS)

Please answer the question given below in no less than 200 words.

1. A monthly family budget is a forecast of income and expenditure of a family in a month. Critically discuss the above statement.
2. Imagine that you have been given the task to observe the food seeking behavior of rats. Would it be best to conduct this in the wild, or in a laboratory situation? Do you think the results will matter?

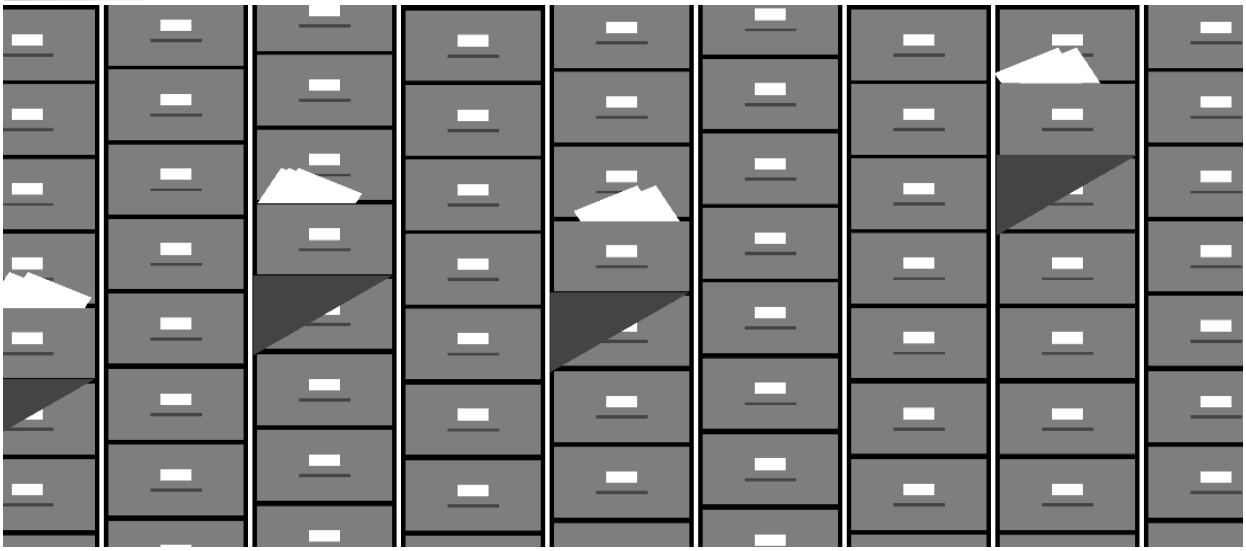
Applied Project

You have been assigned the task to observe the behavior of people working in the parking space of a supermarket. During the observation, you should keep an eye on the maximum number of people working at a time in the parking space. How the employee interacts with the drivers of the vehicles coming in and going out of the parking space. Do they provide any extra assistance to the customers? How well are they able to manage space, so that they can accommodate parking for maximum number of cars at peak shopping hours? You are free to add any observation that you may find interesting.

CHAPTER

4

Randomization



Studying this chapter should enable you to understand:

- Use of surveys to collect data
- Sampling bias
- Confidence interval
- Data collection by sensory devices
- Data from internet

1. Introduction

In the previous chapter, we learned about how we can use the observational study to collect data. The collected data is further analyzed to deduce a conclusion.

In this chapter, we will learn about how we can collect data via mediums like surveys, sensory devices, and the internet. We will also explore a way to increase the accuracy of the results deduced using a confidence interval.

2. Let us do a survey

A survey is a research method used to collect data where the subjects are generally people. In a survey, the process involves asking people for the information through a questionnaire. The outcome of a survey depends heavily on the type of questions asked. The questions should be carefully worded not to hurt the sentiment of the people being surveyed.

Surveys can be composed of two types of questions: open-ended questions and close-ended questions. The respondents can answer open-ended questions in their own words. In close-ended questions, the choice of answers from which to select is fixed and generally provided alongside the question.

Some examples of open-ended questions are:

- Comments/review
- Suggest improvements

Some examples of close-ended questions are:

- Multiple choice
- Yes/No
- A rating scale of 1 to 10
- Emojis

Shown alongside is an example of student feedback survey on their favorite sport for grade XI & XII.

A student needs to select a grade, favorite sport and then describe why he/she likes the selected sport. The response collected from this survey can then be analyzed upon to find meaningful insights.

Fig 4.1 Student survey about favorite sport

3. Sampling Bias

Sampling bias is a type of discrimination in which a sample is collected so that some members of the considered population have a lower or a higher sampling chance than the others. This sample type can be considered a non-random sample as the likelihood of everyone being equally selected is not there. If such a scenario is not accounted for, it will generate wrong results for the phenomenon under study.

In other words, to make the statistic unbiased, sample collection should be random. Thus, all the members of the considered population should have an equal sampling chance.

4. How sure are you?

When we ask someone, "How sure are you?" we try to gauge the level of confidence with which that person is putting forward an observation.

In statistics, the term used to measure the accuracy of a result is called the **confidence interval**.

To understand confidence interval in detail, we first need to understand sampling and sampling error. To find things out about a population of interest, it is common practice to take a sample. A sample is a selection of objects or observations taken from the population of interest. For example, a population can be all mangoes in an orchard at a given time. We wish to know; how heavy the mangoes are. We cannot measure all of them, so we take a sample of some of them and measure them.

Let us first understand what population parameter. A population parameter is a value that describes the characteristics of an entire population, such as the population mean.

The inference is when we conclude the population from the sample. Because the sample is only a selection of objects from the population, it will never be a perfect representation of the population. Separate samples of the same population will give different results, giving rise to sampling error or variation. Thus, there will always be sampling errors.

To sum up, when we estimate a population parameter, it is good practice to give it a confidence interval. A confidence interval communicates how accurate our estimate is likely to be.

Thus, if we put an investigative question like:

What is the mean weight of all the mangoes in the orchard? [4]

For this, we take a sample of mangoes and calculate the sample mean, which is

the best estimate of the population mean.

A confidence interval defines the span in which we are pretty sure the population parameter lies. In this case, the mean weight for all the mangoes in the orchard is the population parameter.

So, in this case, if we consider that the mean weight of mangoes in the orchard is 250 gm and the confidence interval is 20, we can represent it as follows:

Fig 4.2 Mean weight of mangoes in the orchard with confidence interval of 20

Now that we know about confidence interval let us find out what affects a confidence interval's width.

The width of a confidence interval depends upon two things:

1. **The first thing is variations within the population of interest**. If all the population values were almost the same, then we will have low/little variation. Our estimate is going to be close to the actual population. Thus, the confidence interval, in this case, will be small. But a more diverse population will lead to a more diverse sample. Different samples taken from the same population will differ more. We will be less sure that the mean of the sample will be closer to the population mean. Thus, here the confidence interval will be large. So greater dissimilarity in the population leads to a wider confidence interval.

2. **The width of the confidence interval is also affected by the sample size**. With a small sample, we do not have much reference to base our conclusion. Small samples will differ more from one another, leading to a wider confidence interval.
On the other hand, in larger sample size, the effect of a few unusual values is evened out by the other values in the sample. Larger samples will be more like each other. The effective sampling error is reduced with larger samples. When we take larger samples, we have more information and can be surer about our estimates, which leads to a narrower confidence interval.

5. Let us act on a sense

Another method that can be used to collect data is via sensors. This method of data collection requires the least human involvement.

A sensor is a device that identifies and measures the change in input from a physical entity and converts them into signals. These generated signals can then be converted into human-readable displays.

Here is an example of a sensor:

In a mercury-based glass thermometer, the temperature is the input. Depending on the temperature change, the mercury either expands or contracts, causing the level to go up or down on the marked gauge, which is human readable.

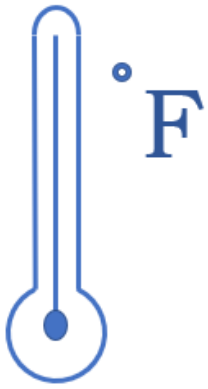


Fig 4.3 Thermometer to collect data based on sense

A sensor can either collect data continuously or whenever a trigger gets activated. Data collection can also be done automatically without any human intervention and following a predefined set of rules.

6. Online Data

The internet can be considered as an ocean of data. There is an uncountable number of websites and web articles on the internet. All of these serve as a rich pool of data. Data can be easily collected from the internet using web data scraping, cleaning up the data, and then analyzing them.

7. Charm of XML

Whenever we collect data for analysis, we first decide upon a subject on which the same needs to be performed. We then go one level deeper to understand the characteristics that need to be observed to perform the analysis.

Shown below is a table highlighting upvotes for different types of pizzas.

Pizza Crust	Upvote
Thin Crust	50
Regular	10
Cheese Burst	30
Deep Dish	20

Thus here, we perform an analysis on pizzas. The study is based on the upvotes pizzas of different pizza crust categories have received.

We can also have a similar kind of table on a web page on the internet. We can store the data shown in these tables on the internet in an XML.

XML stands for Extensible Markup Language. It is a self-descriptive tool to store and transport data on the internet. A simple XML is made up of tags, element names, and element values.

A tag, either opening or closing, is used to mark an element's start or end. Tags are of two types: start tag and end tag. Start tag is created by wrapping the element name between '<' and '>.' An ending tag is created by wrapping the element name between '</' and '>'. The element value is present between the start tag and the end tag.

In XML, each tag is called a node. Each node can have one or more child nodes contained within it.

Thus, if we want to represent the above table as an XML, it will be as shown below:

```
<Pizzas>
    <Pizza>
        <Pizza Crust>Thin Crust</Pizza Crust>
        <Upvote>50</Upvote>
    </Pizza>
    <Pizza>
        <Pizza Crust>Regular</Pizza Crust>
        <Upvote>10</Upvote>
    </Pizza>
    <Pizza>
        <Pizza Crust>Cheese Burst</Pizza Crust>
        <Upvote>30</Upvote>
    </Pizza>
    <Pizza>
        <Pizza Crust>Deep Dish</Pizza Crust>
        <Upvote>20</Upvote>
    </Pizza>
</Pizzas>
```

XML format makes it simple to display data on a web page. Also, converting XML to a data table format helps us visualize our data better.

Thus, we can get an XML to collect data and perform analysis on it.

Recap

- In this chapter, we have learnt about use of surveys to collect data. We also understood how to design the questions in a survey and the different types of questions that may find its place in a survey.
- Introducing biasness while collecting sample will give incorrect results.
- Sometimes, results from an experiment are stated as an approximation. The maximum range possible between the approximated value and the actual value is confidence interval.
- Data can be collected via different mediums from different places. We can collect temperature data via a thermometer, while data on the internet can be collected via xml.

Exercises

Objective Type Questions

1) Which of the following statement is false?
   a) Yes/No is an example of close ended question in a survey.
   b) A rating scale of 1 – 10 is an example of close ended question in a survey.
   c) A multiple-choice option is an example of close ended question in a survey.
   d) Suggesting improvement is an example of close ended question in a survey.

2) Out of the options given below, which one can be selected to be associated with survey research?
   e) The problem of objectivity
   f) The problem of "going native"
   g) The problem of omission
   h) The problem of robustness

3) Mr. X conducted a study of the way restaurant owners granted or refused access to a couple. This is an example of observing behavior in terms of:
   a) Individuals
   b) Incidents
   c) Short time periods
   d) Long time periods

4) The statement "results are accurate within +/-4 p.p., 95% of the time" refers respectively to:
   a) confidence level and confidence interval
   b) confidence interval and confidence level
   c) margin of error and margin of confidence
   d) sample interval and confidence level

5) Sampling error can be reduced by:
   a) correcting a faulty sample frame
   b) increasing response rates
   c) increasing the sample size
   d) reducing incomplete surveys

6) Greater dissimilarity within population of interest will:
   a) Increase the confidence interval
   b) Decrease the confidence interval
   c) Will have no impact on confidence interval
   d) None of the above

7) Which of the following is a method of data collection by sensing?
   a) Use of speed guns to measure the speed of vehicles by traffic police.
   b) Performing surveys to calculate population of the country.
   c) Performing experiments and observing the results.
   d) None of the above

8) What is the best format in which online data should be collected perform analysis?
   a) RTF
   b) DOCX
   c) XML
   d) CSV

Standard Questions

1) What is a survey? What are the things to keep in mind while creating a survey?
2) What are the different types of questions that a survey can contain? Provide two examples for each type of question.
3) What is sampling bias? Is it reasonable to have a sampling bias? Provide an example to support your answer.
4) Given below are instances of biased survey questions. Point out the biasness involved and try rephrasing the questions so that the subject responding to them can generate meaningful response.
   a) How amazing was your experience with our customer service team?
   b) What problems did you have with the launch of this new product?
   c) How do we compare to our competitors?
   d) Do you always use product X for your cleaning needs?
5) What is a confidence interval? What are the different factors affecting the values of a confidence interval for a given population?
6) Explain with an example how sensors are used widely in the field of healthcare to collect data and monitor patients' health conditions.

Higher Order Thinking Skills (HOTS)

1) India is a nation where most people are incredibly fond of watching cricket as a sport. In order to better strategize against the opponent, each team analyses the strengths and weaknesses of the opponent, a lot of data is analyzed off the field. Explain in detail, how this is done.

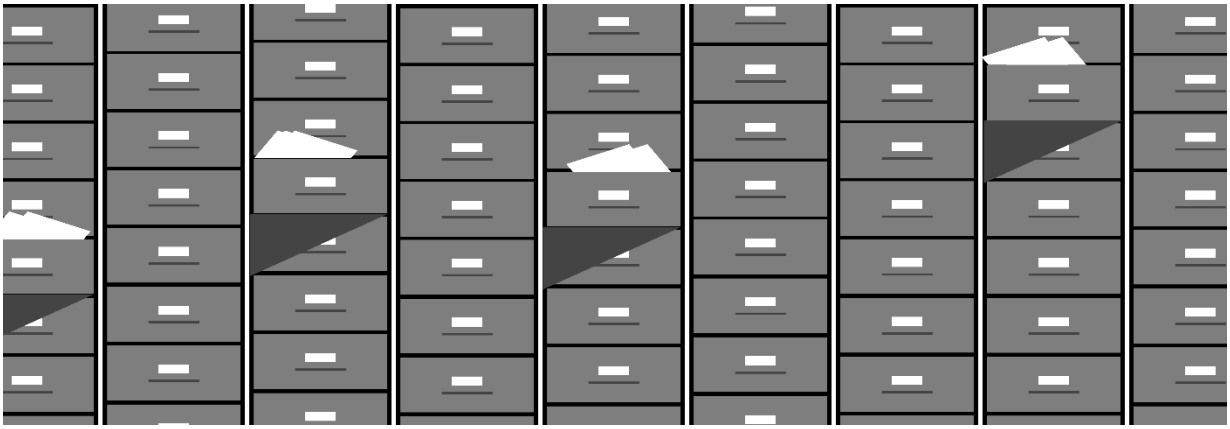
Applied Project

Consider a situation where a person gets admitted to a hospital for treatment. In such situations, information is being collected from the patient using various ways. Explain in detail the various ways data is being collected from the patient. Once the data has been collected, represent the data first in tabular format and then as an XML.

CHAPTER

5

Introduction to R Studio



Studying this chapter should enable you to understand:

- Orientation with R Studio
- Coding for data science using R Studio

1. Introduction

In the previous chapter, we learned about collecting data from surveys, sensors, and the internet. We also explained in detail the concept of a confidence interval.

In this chapter, we will learn about R Studio and coding for data science using R Studio.

2. Orientation with R Studio

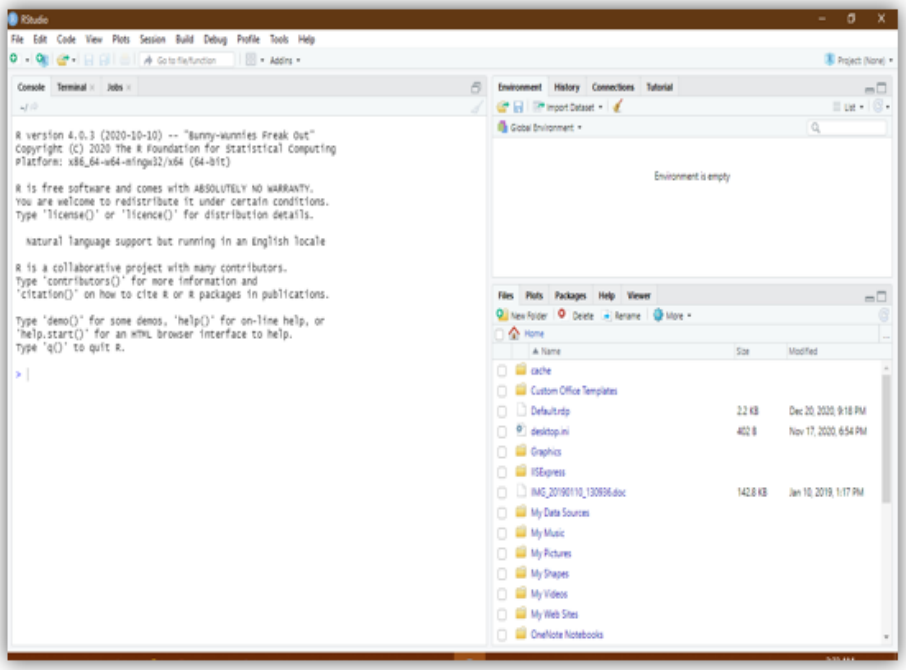
To download R Studio, navigate to the link given below:

https://rstudio.com/products/rstudio/download/

We need to download the R Studio installer for windows and install it on the windows machine.

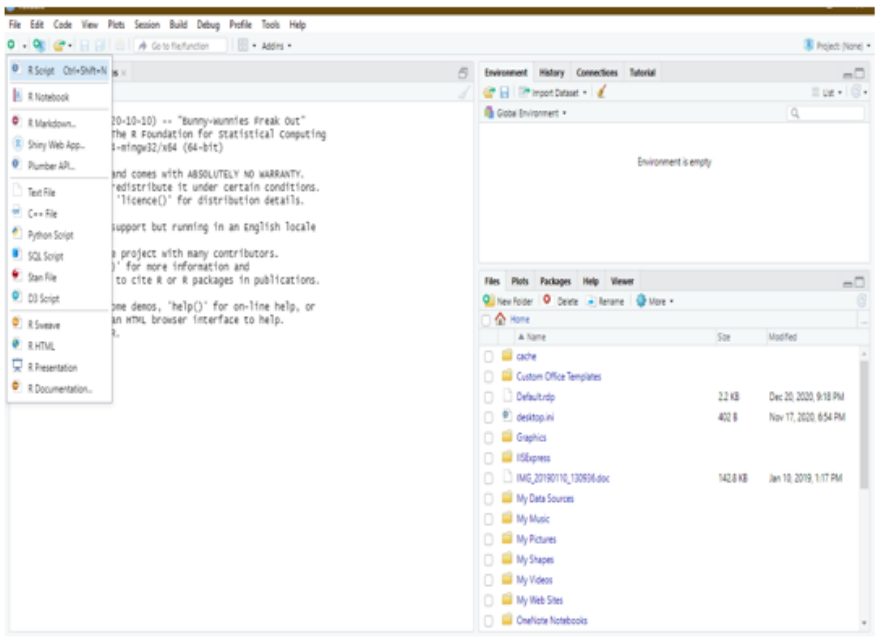
Once the R Studio gets installed successfully on the windows machine, we need to open it to start working.

When the R-Studio is opened for the first time, we get an interface, as shown below:

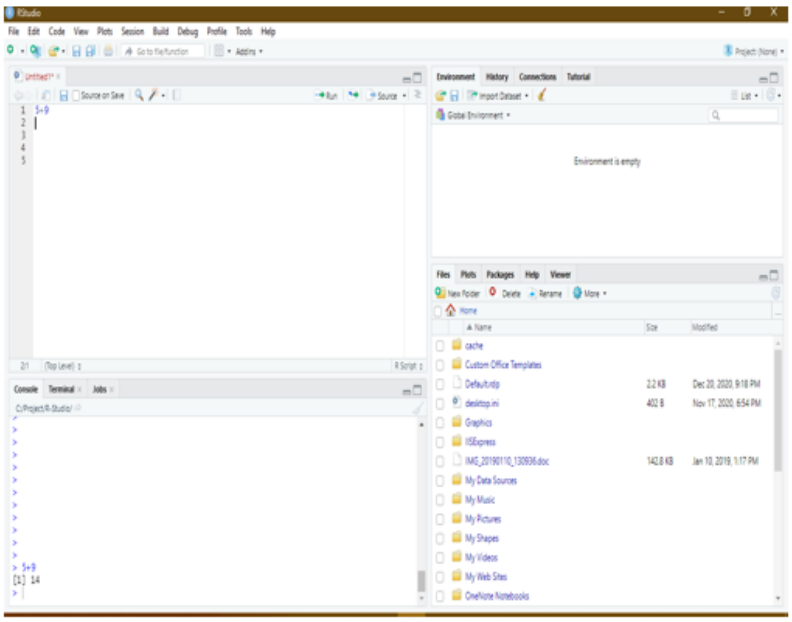


On the left-hand side, there is a window having tabs **Console**, **Terminal** & **Jobs**.

On the right-hand side, there are two windows. The window on the top has the following tabs – **Environment, History, Connections, Tutorials**. The window on the bottom has the following tabs – **Files, Plots, Packages, Help, Viewer**.

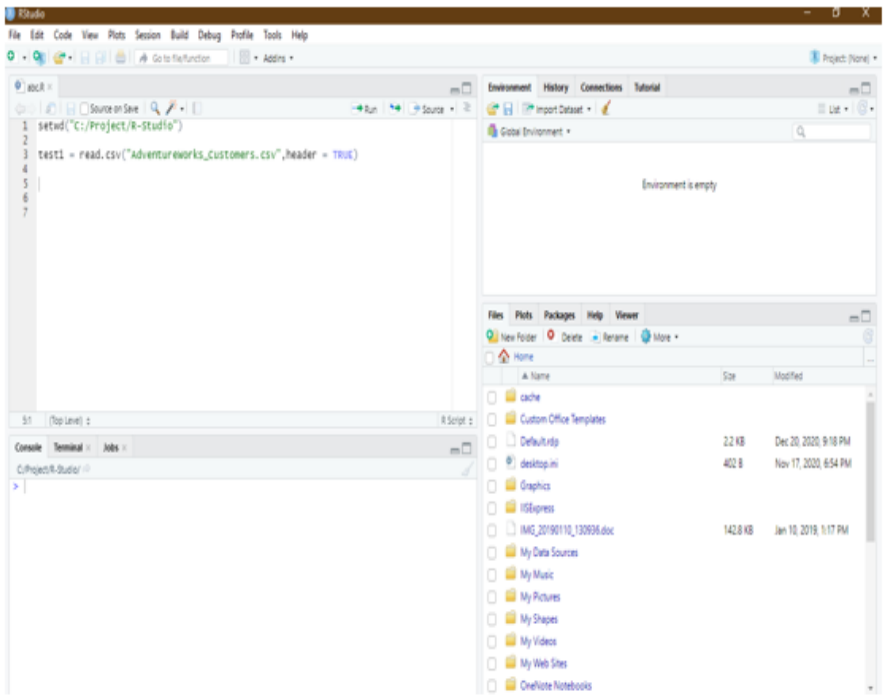


For more of the practical work, we need a fourth window to write script. To do so click on the **+** below File and select **R Script.** This will open a new window on the top left-hand side.



In the new script window which opens, we type a simple math expression and then press **Run**. The result will show up in the console.

We can load a .csv file from a directory and can see the contents in R-Studio like below:



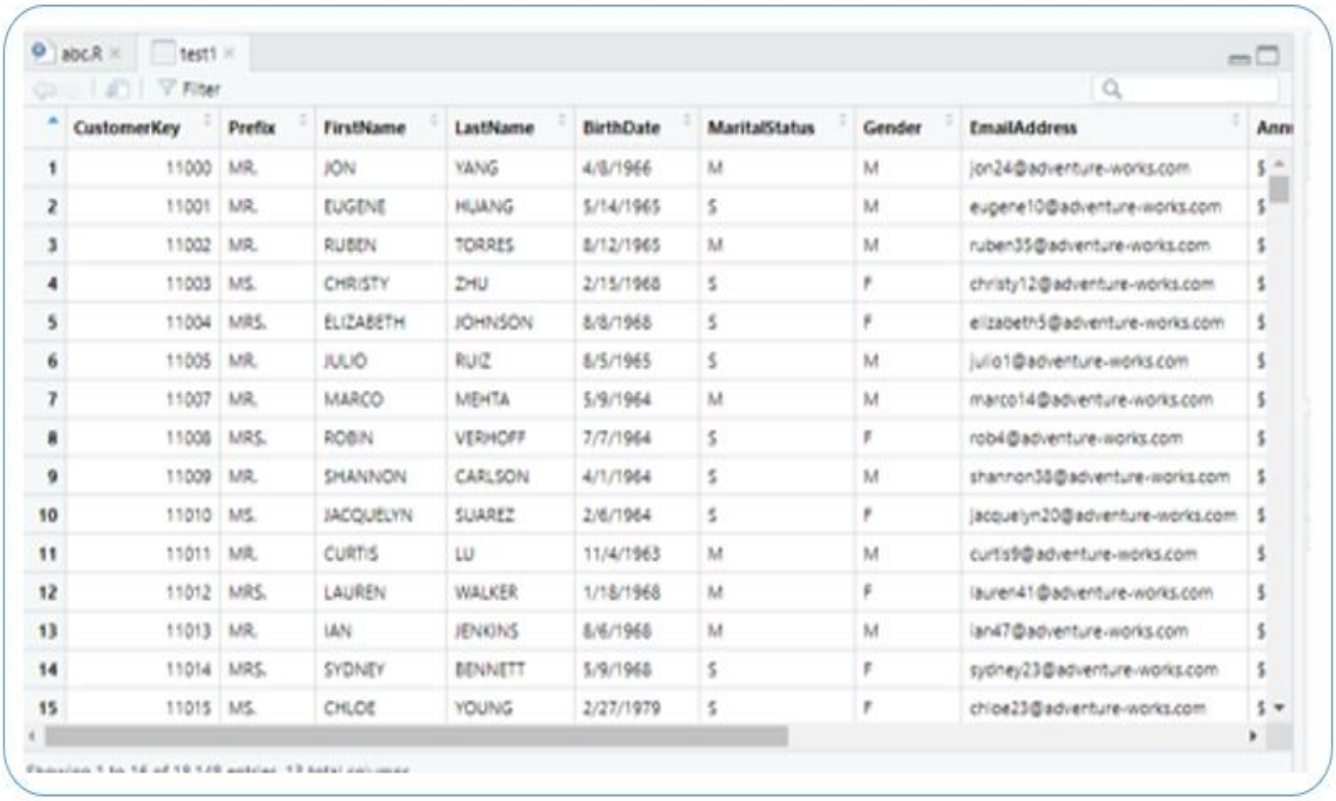
Step1: First, we set the working directory.

Step2: Then we read a csv file kept in that directory.

Step3: We click on the Source button.

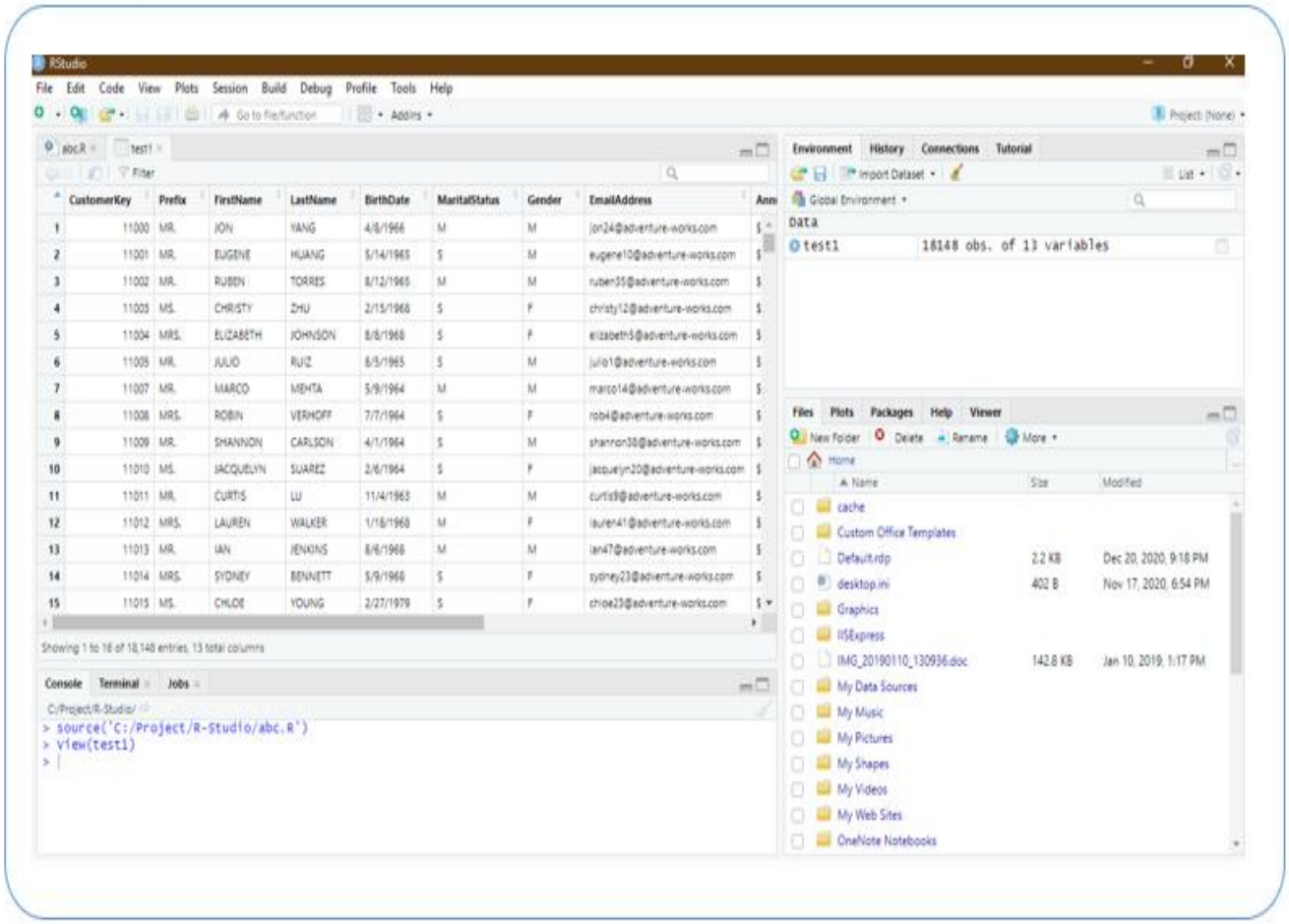
Step4: On clicking the Source button, variable test1 shows up in the environment window.

Step5: On clicking on the test1 variable in the environment window, a new window opens which displays the contents of the csv file.


abc.R × test1 ×

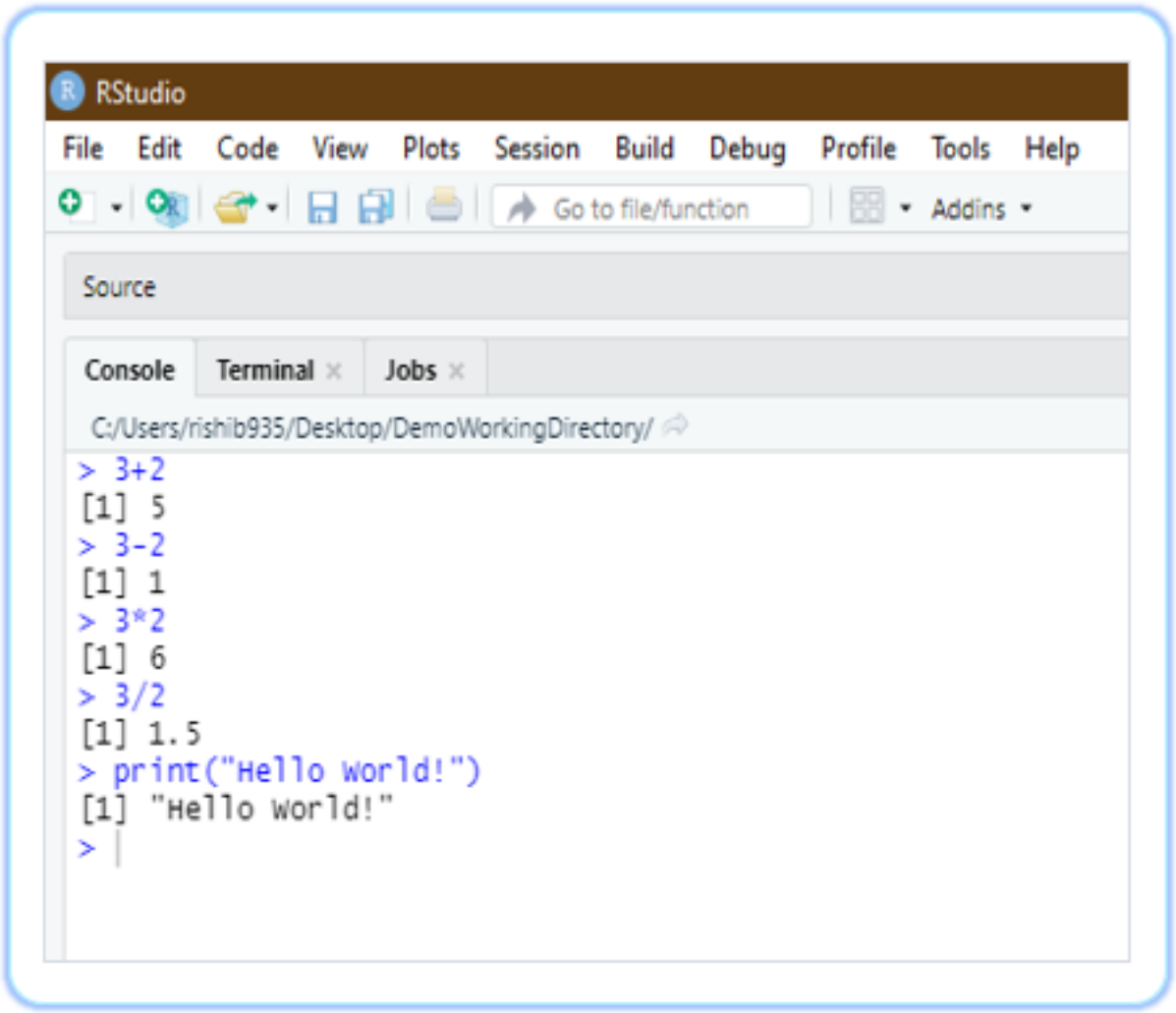
Filter

	CustomerKey	Prefix	FirstName	LastName	BirthDate	MaritalStatus	Gender	EmailAddress	Ann
1	11000	MR.	JON	YANG	4/8/1966	M	M	jon24@adventure-works.com	$
2	11001	MR.	EUGENE	HUANG	5/14/1965	S	M	eugene10@adventure-works.com	$
3	11002	MR.	RUBEN	TORRES	8/12/1965	M	M	ruben35@adventure-works.com	$
4	11003	MS.	CHRISTY	ZHU	2/15/1968	S	F	christy12@adventure-works.com	$
5	11004	MRS.	ELIZABETH	JOHNSON	8/8/1968	S	F	elizabeth5@adventure-works.com	$
6	11005	MR.	JULIO	RUIZ	8/5/1965	S	M	julio1@adventure-works.com	$
7	11007	MR.	MARCO	MEHTA	5/9/1964	M	M	marco14@adventure-works.com	$
8	11008	MRS.	ROBIN	VERHOFF	7/7/1964	S	F	rob4@adventure-works.com	$
9	11009	MR.	SHANNON	CARLSON	4/1/1964	S	M	shannon38@adventure-works.com	$
10	11010	MS.	JACQUELYN	SUAREZ	2/6/1964	S	F	jacquelyn20@adventure-works.com	$
11	11011	MR.	CURTIS	LU	11/4/1963	M	M	curtis9@adventure-works.com	$
12	11012	MRS.	LAUREN	WALKER	1/18/1968	M	F	lauren41@adventure-works.com	$
13	11013	MR.	IAN	JENKINS	8/6/1968	M	M	ian47@adventure-works.com	$
14	11014	MRS.	SYDNEY	BENNETT	5/9/1968	S	F	sydney23@adventure-works.com	$
15	11015	MS.	CHLOE	YOUNG	2/27/1979	S	F	chloe23@adventure-works.com	$

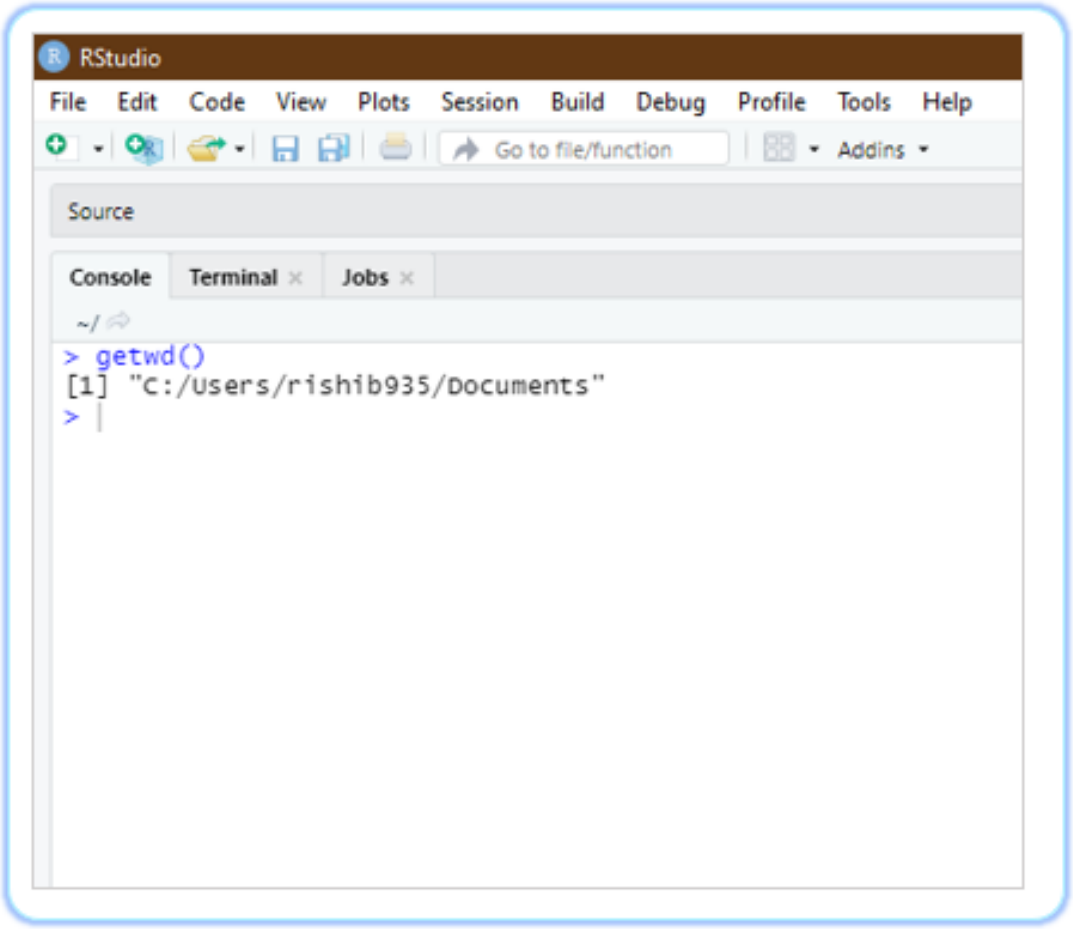

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
abcR
test1
Filter
CustomerKey Prefix FirstName LastName BirthDate MaritalStatus Gender EmailAddress
11000 MR. JON YANG 4/8/1966 M M jon24@adventure-works.com
11001 MR. EUGENE HUANG 5/14/1965 S M eugene10@adventure-works.com
11002 MR. RUBEN TORRES 8/12/1965 M M ruben35@adventure-works.com
11003 MS. CHRISTY ZHU 2/15/1968 S F christy12@adventure-works.com
11004 MRS. ELIZABETH JOHNSON 8/8/1968 S F elizabeth5@adventure-works.com
11005 MR. JULIO RUIZ 8/5/1965 S M julio1@adventure-works.com
11007 MR. MARCO MEHTA 5/9/1964 M M marco14@adventure-works.com
11008 MRS. ROBIN VERHOFF 7/7/1964 S F robin@adventure-works.com
11009 MR. SHANNON CARLSON 4/1/1964 S M shannon38@adventure-works.com
11010 MS. JACQUELYN SUAREZ 2/6/1964 S F jacquelyn20@adventure-works.com
11011 MR. CURTIS LU 11/4/1963 M M curtis9@adventure-works.com
11012 MRS. LAUREN WALKER 1/18/1968 M F lauren41@adventure-works.com
11013 MR. IAN JENKINS 8/6/1968 M M ian47@adventure-works.com
11014 MRS. SYDNEY BENNETT 5/9/1968 S F sydney23@adventure-works.com
11015 MS. CHLOE YOUNG 2/27/1979 S F chloe23@adventure-works.com
Showing 1 to 16 of 18,148 entries, 13 total columns
Console Terminal Jobs
C:/Project/R-Studio/
> source('C:/Project/R-Studio/abc.R')
> View(test1)
Environment History Connections Tutorial
Import Dataset
Global Environment
Data
test1 18148 obs. of 13 variables
Files Plots Packages Help Viewer
New Folder Delete Rename More
Home
Name Size Modified
cache
Custom Office Templates
Default.rdp 2.2 KB Dec 20, 2020, 9:18 PM
desktop.ini 402 B Nov 17, 2020, 6:54 PM
Graphics
IISExpress
IMG_20190110_130936.doc 142.8 KB Jan 10, 2019, 1:17 PM
My Data Sources
My Music
My Pictures
My Shapes
My Videos
My Web Sites
OneNote Notebooks

The console window in RStudio is the place where we can tell it what to do and it will show the results of a command. We can type commands directly into the console, but the drawback is that they will be forgotten when we close the session.

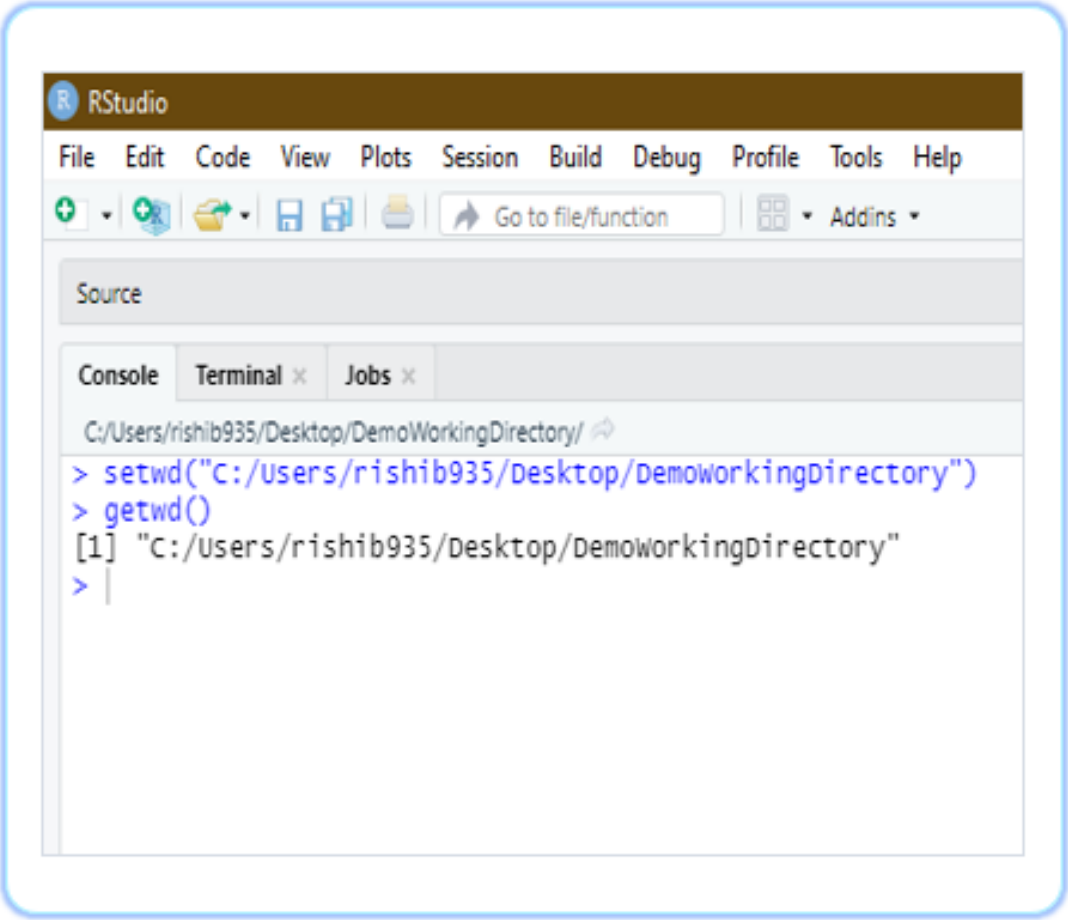
Some examples of simple commands executed via the console in RStudio as given below:



Every time RStudio is opened, it goes to a working directory. We can know the current working directory in RStudio using the command getwd() in the console.



We can change the working directory to a folder of our choice. To do so, we use the setwd() function. The directory path of the directory, which we want to set as working directory is passed as a string parameter in the function. The directory path which is passed as a parameter can either be a relative path or an absolute path.



Vector in R programming

In R, a sequence of elements that share the same data type is known as a vector. Vectors are the most basic data objects. There are six basic vectors – **logical, integer, double, complex, character, and raw**. We also call these basic vectors atomic vectors.

When a person writes just one value in R, it becomes a vector of length one and belongs to one of the above-stated vector types. Such a vector is called a **single-element vector**.

```r
# Atomic vector of type character.
print("Science");

# Atomic vector of type double.
print(62.6)

# Atomic vector of type integer.
print(98L)

# Atomic vector of type logical.
print(FALSE)

# Atomic vector of type complex.
print(5-2i)

# Atomic vector of type raw.
print(charToRaw('data'))
```

Example of different vectors of length 1 in R studio has been shown here

Just like a single element vector, we also have multiple element vector.

We can create a multiple elements vector with numeric data using a colon operator.

```
> print("Science");
[1] "Science"
>
> # Atomic vector of type double.
> print(62.6)
[1] 62.6
>
> # Atomic vector of type integer.
> print(98L)
[1] 98
>
> # Atomic vector of type logical.
> print(FALSE)
[1] FALSE
>
> # Atomic vector of type complex.
> print(5-2i)
[1] 5-2i
>
> # Atomic vector of type raw.
> print(charToRaw('data'))
[1] 64 61 74 61
>
```

When the above code is executed in the R Studio, it generates the following output

```r
# We can create a sequence from 1 to 15
v <- 1:15
print(v)

# We can  create a sequence from 2.2 to 8.2.
v <- 2.2:8.2
print(v)

# If the last element mentioned does not belong to the sequence then it is discarded.
v <- 2.6:12.8
print(v)
```

Creating multiple elements vector with numeric data using a colon operator

Using the sequence operator, we can create a vector with elements between two numbers, the values for which increments by a numerical figure.

```
> # We can create a sequence from 1 to 15
> v <- 1:15
> print(v)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
>
> # We can  create a sequence from 2.2 to 8.2.
> v <- 2.2:8.2
> print(v)
[1] 2.2 3.2 4.2 5.2 6.2 7.2 8.2
>
> # If the last element mentioned does not belong to the sequence then it is discarded.
> v <- 2.6:12.8
> print(v)
 [1]  2.6  3.6  4.6  5.6  6.6  7.6  8.6  9.6 10.6 11.6 12.6
>
```

When the above code is executed in the R Studio, it generates the following output

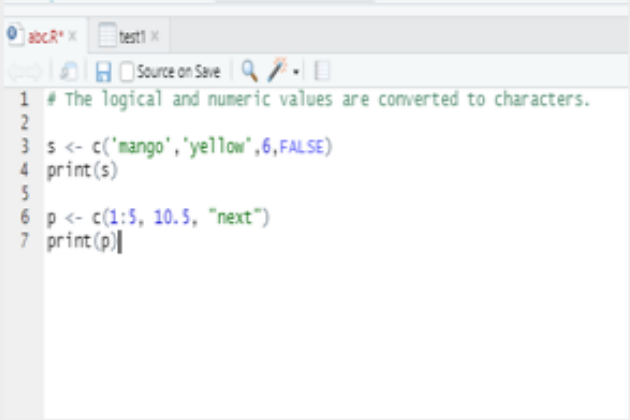
```
x <- seq(7 , 11 , by = 0.5)
print(x)
```

To display all numbers from 7 till 11 at an interval of 0.5 each.

```
> x <- seq(7 , 11 , by = 0.5)
> print(x)
[1]  7.0  7.5  8.0  8.5  9.0  9.5 10.0 10.5 11.0
>
```

Results from the code run above

Another way of creating a vector is to use the **c** function. Here, the default method combines the arguments provided to form a vector. Here, all the arguments are forced to a common type, which is the returned value type. The return type is determined from the highest type of the components in the hierarchy **expression > list > character > complex > double > integer > logical > raw > NULL**.



To create a vector using the arguments provided.

```
Console  Terminal  Jobs
C:/Project/R-Studio/
> # The logical and numeric values are converted to characters.
>
> s <- c('mango','yellow',6,FALSE)
> print(s)
[1] "mango"  "yellow" "6"      "FALSE"
>
> p <- c(1:5, 10.5, "next")
> print(p)
[1] "1"    "2"    "3"    "4"    "5"    "10.5" "next"
>
```

Results from the code run above

In order to access the elements in a vector, indexing is being used. The [] brackets are used for indexing. Indexing starts with position 1. Providing a negative value in the index drops the element from the result. We can also use **TRUE/FALSE** or **0 and 1** for indexing.

```
abc.R*  test1
Source on Save
1  # Accessing vector elements using position.
2
3  t <- c("Jan","Feb","Mar","Apr","May","Jun","Jul")
4  u <- t[c(1,4,6)]
5  print(u)
6
7  # Accessing vector elements using logical indexing.
8  v <- t[c(TRUE,FALSE,TRUE,FALSE,FALSE,TRUE,FALSE)]
9  print(v)
10
11 # Accessing vector elements using negative indexing.
12 x <- t[c(-1,-4)]
13 print(x)
14
15 # Accessing vector elements using 0/1 indexing.
16 y <- t[c(1,0,0,1,0,0,1)]
17 print(y)
```

Example of operation with vectors using indexing.

```
Console   Terminal   Jobs
C:/Project/R-Studio/
> # Accessing vector elements using position.
>
> t <- c("Jan","Feb","Mar","Apr","May","Jun","Jul")
> u <- t[c(1,4,6)]
> print(u)
[1] "Jan" "Apr" "Jun"
>
> # Accessing vector elements using logical indexing.
> v <- t[c(TRUE,FALSE,TRUE,FALSE,FALSE,TRUE,FALSE)]
> print(v)
[1] "Jan" "Mar" "Jun"
>
> # Accessing vector elements using negative indexing.
> x <- t[c(-1,-4)]
> print(x)
[1] "Feb" "Mar" "May" "Jun" "Jul"
>
> # Accessing vector elements using 0/1 indexing.
> y <- t[c(1,0,0,1,0,0,1)]
> print(y)
[1] "Jan" "Jan" "Jan"
>
```

Results from the code run above

Arithmetic operations like addition, subtraction, multiplication & division can also be performed on vectors. For performing arithmetic operations, the two vectors must be of the same length. The result generated post performing the arithmetic operation is also a vector.

In the example shown below, we declare two vectors v1 & v2, and then perform arithmetic operations like addition, subtraction, multiplication, and division.

```r
# Create two vectors.
v1 <- c(1,6,4,9,0,12)
v2 <- c(5,12,0,7,5,3)

# Vector addition operation.
add.result <- v1+v2
print(add.result)

# Vector subtraction operation.
sub.result <- v1-v2
print(sub.result)

# Vector multiplication operation.
multi.result <- v1*v2
print(multi.result)

# Vector division operation.
divi.result <- v1/v2
print(divi.result)
```

Example of arithmetic operation with vectors

Arithmetic operations performed on the two vectors also result in a vector.

```
> # Create two vectors.
> v1 <- c(1,6,4,9,0,12)
> v2 <- c(5,12,0,7,5,3)
>
> # Vector addition operation.
> add.result <- v1+v2
> print(add.result)
[1]  6 18  4 16  5 15
>
> # Vector subtraction operation.
> sub.result <- v1-v2
> print(sub.result)
[1] -4 -6  4  2 -5  9
>
> # Vector multiplication operation.
> multi.result <- v1*v2
> print(multi.result)
[1]  5 72  0 63  0 36
>
> # Vector division operation.
> divi.result <- v1/v2
> print(divi.result)
[1] 0.200000 0.500000      Inf 1.285714 0.000000 4.000000
>
```

Results from the code run above

List in R programming

A list in R is a type of R object which contains different types of elements like - numbers, vectors, strings, and another list within it. A list can also contain a function or a matrix as its elements.

To create a list, we use the **list()** function. Shown below is an example to create a list using strings, numbers, vectors, and logical values.

```
# Create a list containing strings, numbers, vectors and a logical
# values.
list_data <- list("Blue", "Orange", c(15,27,62), FALSE, 67.48, 128.3)
print(list_data)
```

Example of a code to create a list

```
> # Create a list containing strings, numbers, vectors and a logical
> # values.
> list_data <- list("Blue", "Orange", c(15,27,62), FALSE, 67.48, 128.3)
> print(list_data)
[[1]]
[1] "Blue"

[[2]]
[1] "Orange"

[[3]]
[1] 15 27 62

[[4]]
[1] FALSE

[[5]]
[1] 67.48

[[6]]
[1] 128.3
```

Results from the code run above

Naming elements in a list

Names can be given to list elements, and they can be accessed using the same.

Shown below is an example of assigning names to the elements in the list.

```r
# Create a list containing a vector, a matrix and a list.
list_data <- list(c("Saturday","Sunday"),
                  matrix(c(5,4,2,9,-7,6), nrow = 2),
                  list("blue",TRUE))

# Give names to the elements in the list.
names(list_data) <- c("Weekend Days",
                      "Example_Matrix",
                      "Example Inner list")

# Show the list.
print(list_data)
```

Example of assigning names to elements in a list

```
C:/Project/R-Studio/
> # Create a list containing a vector, a matrix and a list.
> list_data <- list(c("Saturday","Sunday"),
+                   matrix(c(5,4,2,9,-7,6), nrow = 2),
+                   list("blue",TRUE))
>
> # Give names to the elements in the list.
> names(list_data) <- c("Weekend Days",
+                       "Example_Matrix",
+                       "Example Inner list")
>
> # Show the list.
> print(list_data)
$`Weekend Days`
[1] "Saturday" "Sunday"

$Example_Matrix
     [,1] [,2] [,3]
[1,]    5    2   -7
[2,]    4    9    6

$`Example Inner list`
$`Example Inner list`[[1]]
[1] "blue"

$`Example Inner list`[[2]]
[1] TRUE
```

Results from the code run above

Accessing elements in a list

Elements in a list can be accessed using the index of the element in the list. In case the list is a named list, it can also be accessed using the names.

To give a demonstration, let us use the list shown in the above example:

```r
# Create a list containing a vector, a matrix and a list.
list_data <- list(c("Saturday","Sunday"),
                  matrix(c(5,4,2,9,-7,6), nrow = 2),
                  list("blue",TRUE))

# Give names to the elements in the list.
names(list_data) <- c("Weekend Days",
                      "Example_Matrix",
                      "Example Inner list")

# Access the first element of the list.
print(list_data[1])

# Access the thrid element. As it is also a list,
#all its elements will be printed.
print(list_data$'Example Inner list')
```

Example of accessing elements in a list

```
> # Create a list containing a vector, a matrix and a list.
> list_data <- list(c("Saturday","Sunday"),
+                   matrix(c(5,4,2,9,-7,6), nrow = 2),
+                   list("blue",TRUE))
>
> # Give names to the elements in the list.
> names(list_data) <- c("Weekend Days",
+                       "Example_Matrix",
+                       "Example Inner list")
>
> # Access the first element of the list.
> print(list_data[1])
$`Weekend Days`
[1] "Saturday" "Sunday"

>
> # Access the thrid element. As it is also a list,
> #all its elements will be printed.
> print(list_data$'Example Inner list')
[[1]]
[1] "blue"

[[2]]
[1] TRUE
```

Results from the code run above

Manipulating elements in a list

In a list in R, we can add, delete or update elements. The addition or deletion of the elements can only be done at the end of the list. However, an update can be performed on any element in the list.

As a demonstration, let us use the list shown in the above example:

abc.R*

Source on Save

```
# Create a list containing a vector, a matrix and a list.
list_data <- list(c("Saturday","Sunday"),
                  matrix(c(5,4,2,9,-7,6), nrow = 2),
                  list("blue",TRUE))

# Give names to the elements in the list.
names(list_data) <- c("Weekend Days",
                      "Example_Matrix",
                      "Example Inner list")


# Any element is always added at the end of the list.

list_data[4] <- "Test element"
print(list_data[4])

# Any element is always removed from the end of the list.

list_data[4] <- NULL
print(list_data[4])

# Update can be done on any element in the list.
# Update the 2nd Element.
list_data[2] <- "Matrix updated by string"
print(list_data[2])
```

Example of manipulating elements in a list

Results from the code run above

Matrices in R programming

In R, matrices are an extension of the numeric or character vectors. In other words, they are atomic vectors arranged in a two-dimensional rectangular layout. Thus, matrix being an atomic vector extension, its elements must be of same data type.

To create a matrix in R, we use the **matrix()** function.

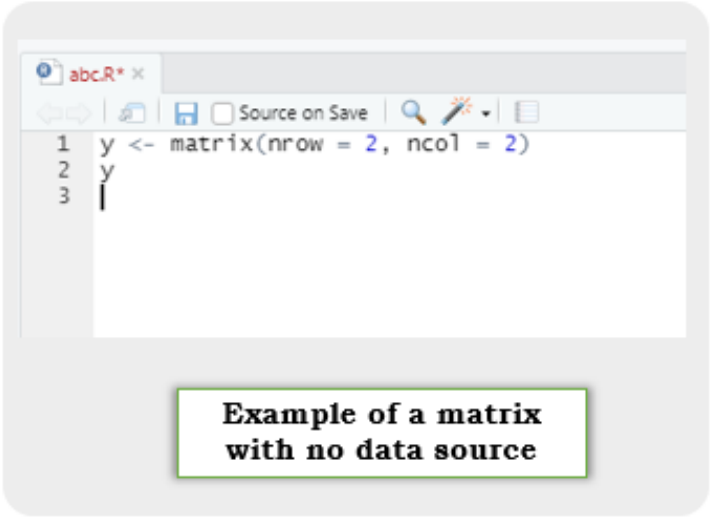
The syntax for creating a matrix in R is:

matrix(data, nrow, ncol, byrow, dimnames)

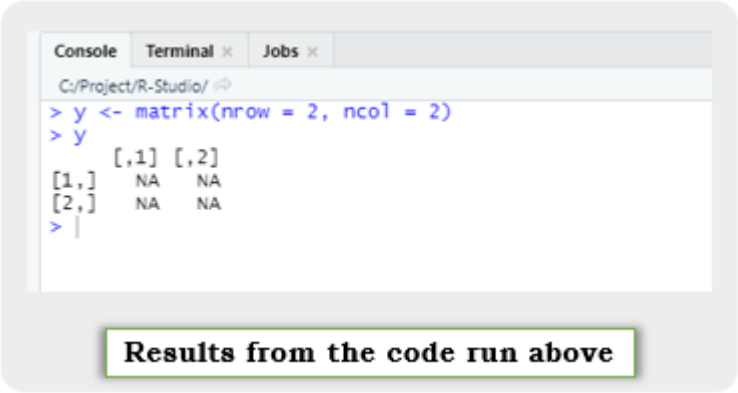
The parameters used can be described as follows:

- **data**: the input vector which becomes the data elements of the matrix.
- **nrow**: number of rows to be created.
- **ncol**: number of columns to be created.
- **byrow** represents a logical clue. When set to TRUE, then the elements in input vector are organized by row.
- **dimname** is the names assigned to the rows and columns.

Shown below is an example of a matrix where no data source is provided.



Example of a matrix with no data source



Results from the code run above

Shown here is an example of creating a matrix taking a vector of numbers as input

abc.R*

Source on Save | Run

```
1  # Elements are arranged sequentially by row.
2  M <- matrix(c(2:13), nrow = 3, byrow = TRUE)
3  print(M)
4
5  # Elements are arranged sequentially by column.
6  N <- matrix(c(2:13), nrow = 3, byrow = FALSE)
7  print(N)
8
9  # Define the column and row names.
10 rownames = c("row1", "row2", "row3")
11 colnames = c("col1", "col2", "col3", "col4")
12
13 P <- matrix(c(2:13), nrow = 3, byrow = TRUE,
14              dimnames = list(rownames, colnames))
15 print(P)
```

Example of a matrix taking vector of numbers as input

Console | Terminal | Jobs

C:/Project/R-Studio/

```
> # Elements are arranged sequentially by row.
> M <- matrix(c(2:13), nrow = 3, byrow = TRUE)
> print(M)
     [,1] [,2] [,3] [,4]
[1,]    2    3    4    5
[2,]    6    7    8    9
[3,]   10   11   12   13
>
> # Elements are arranged sequentially by column.
> N <- matrix(c(2:13), nrow = 3, byrow = FALSE)
> print(N)
     [,1] [,2] [,3] [,4]
[1,]    2    5    8   11
[2,]    3    6    9   12
[3,]    4    7   10   13
>
> # Define the column and row names.
> rownames = c("row1", "row2", "row3")
> colnames = c("col1", "col2", "col3", "col4")
>
> P <- matrix(c(2:13), nrow = 3, byrow = TRUE,
+             dimnames = list(rownames, colnames))
> print(P)
     col1 col2 col3 col4
row1    2    3    4    5
row2    6    7    8    9
row3   10   11   12   13
>
```

Results from the code run above

How to access elements in a matrix

Elements of a matrix can be fetched by using the column and row index of the element.

Shown below is a code snippet that illustrates how we can access different elements in an array.

abc.R*

```
1  # Define the column and row names.
2  rownames = c("row1", "row2", "row3")
3  colnames = c("col1", "col2", "col3", "col4")
4
5  # Create the matrix.
6  P <- matrix(c(2:13), nrow = 3, byrow = TRUE,
7              dimnames = list(rownames, colnames))
8
9  # Access the element at 3rd column and 1st row.
10 print(P[1,3])
11
12 # Access the element at 2nd column and 4th row.
13 print(P[3,2])
14
15 # Access only the  2nd row.
16 print(P[2,])
17
18 # Access only the 3rd column.
19 print(P[,3])
20
```

Example of how to access elements in a matrix

Console | Terminal | Jobs

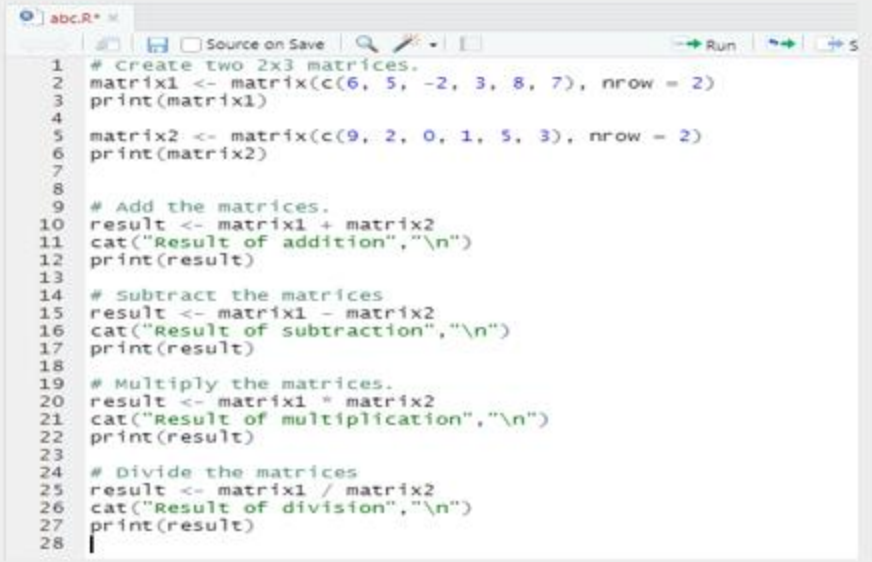
C:/Project/R-Studio/

```
> # Define the column and row names.
> rownames = c("row1", "row2", "row3")
> colnames = c("col1", "col2", "col3", "col4")
>
> # Create the matrix.
> P <- matrix(c(2:13), nrow = 3, byrow = TRUE,
+             dimnames = list(rownames, colnames))
>
> # Access the element at 3rd column and 1st row.
> print(P[1,3])
[1] 4
>
> # Access the element at 2nd column and 4th row.
> print(P[3,2])
[1] 11
>
> # Access only the  2nd row.
> print(P[2,])
col1 col2 col3 col4
   6    7    8    9
>
> # Access only the 3rd column.
> print(P[,3])
row1 row2 row3
   4    8   12
>
```

Results from the code run above

Arithmetic operations such as addition, subtraction, multiplication & division can also be performed on matrices. For performing arithmetic operations, the two matrices must be of the same dimensions. The result generated post performing the arithmetic operation is also a matrix.

Shown below is an example where we declare two matrices matrix1 & matrix2 and then perform arithmetic operations like addition, subtraction, multiplication, division on them.



Example of arithmetic operation in a matrix

```
> # Create two 2x3 matrices.
> matrix1 <- matrix(c(6, 5, -2, 3, 8, 7), nrow = 2)
> print(matrix1)
     [,1] [,2] [,3]
[1,]    6   -2    8
[2,]    5    3    7
>
> matrix2 <- matrix(c(9, 2, 0, 1, 5, 3), nrow = 2)
> print(matrix2)
     [,1] [,2] [,3]
[1,]    9    0    5
[2,]    2    1    3
>
>
> # Add the matrices.
> result <- matrix1 + matrix2
> cat("Result of addition","\n")
Result of addition
> print(result)
     [,1] [,2] [,3]
[1,]   15   -2   13
[2,]    7    4   10
>
> # Subtract the matrices
> result <- matrix1 - matrix2
> cat("Result of subtraction","\n")
Result of subtraction
> print(result)
     [,1] [,2] [,3]
[1,]   -3   -2    3
[2,]    3    2    4
>
> # Multiply the matrices.
> result <- matrix1 * matrix2
> cat("Result of multiplication","\n")
Result of multiplication
> print(result)
     [,1] [,2] [,3]
[1,]   54    0   40
[2,]   10    3   21
>
> # Divide the matrices
> result <- matrix1 / matrix2
> cat("Result of division","\n")
Result of division
> print(result)
          [,1] [,2]     [,3]
[1,] 0.6666667 -Inf 1.600000
[2,] 2.5000000    3 2.333333
>
```

Results from the code run above

Arithmetic operations performed on the two matrices also result in a matrix.

We can also merge many lists into one list. Merging can be done by placing the lists inside a c() function or list() function.

Shown below is an example of two lists being combined into one.

abc.R*

Source on Save

```
1  # Create two lists.
2  list1 <- list(1,2,3)
3  list2 <- list("Sun","Mon","Tue")
4
5  # Merge the two lists.
6  merged.list = c(list1,list2)
7
8  # print the merged list
9  print(merged.list)
10
```

Example of merging two list into one

Console | Terminal | Jobs

C:/Project/R-Studio/

```
> # Create two lists.
> list1 <- list(1,2,3)
> list2 <- list("Sun","Mon","Tue")
>
> # Merge the two lists.
> merged.list = c(list1,list2)
>
> # print the merged list
> print(merged.list)
[[1]]
[1] 1

[[2]]
[1] 2

[[3]]
[1] 3

[[4]]
[1] "Sun"

[[5]]
[1] "Mon"

[[6]]
[1] "Tue"
```

Results from the code run above

Transforming list to vector

We can transform a list into a vector. By doing so, we can perform further manipulation on the elements of the vector. Once a list is being converted to a vector, we can perform all the arithmetic operations possible on vectors. To convert a list into a vector, we use the **unlist()** function. This function takes a list as input and generates a vector as output.

Shown below is an example to convert lists into vectors and perform addition on them.

abc.R*

Source on Save

```
1  # Create lists.
2  list1 <- list(5:9)
3  list2 <-list(20:24)
4
5
6  # Convert the lists to vectors.
7  v1 <- unlist(list1)
8  v2 <- unlist(list2)
9
10 print(v1)
11 print(v2)
12
13 # Now add the vectors
14 result <- v1+v2
15 print(result)
```

Example of converting list to vector

Console Terminal Jobs

C:/Project/R-Studio/

```
> # Create lists.
> list1 <- list(5:9)
> list2 <-list(20:24)
>
>
> # Convert the lists to vectors.
> v1 <- unlist(list1)
> v2 <- unlist(list2)
>
> print(v1)
[1] 5 6 7 8 9
> print(v2)
[1] 20 21 22 23 24
>
> # Now add the vectors
> result <- v1+v2
> print(result)
[1] 25 27 29 31 33
>
```

Results from the code run above

Arrays in R programming

Arrays are the R data objects in which we can store data in more than two dimensions. So, if we create an array of dimensions (4,5,2), it will create two rectangular matrices, each with four rows and five columns. Arrays can store only data types.

An array is created using the **array()** function. The array() function takes vectors as input and uses the dim parameter values to create an array.

Shown below is a simple example of an array

```r
# Create two vectors of different lengths.
vector1 <- c(6,4,1)
vector2 <- c(9,12,15,16,14,11)

# Take these vectors as input to the array.
result <- array(c(vector1,vector2),dim = c(3,3,2))
print(result)
```

Example of an array

```
> # Create two vectors of different lengths.
> vector1 <- c(6,4,1)
> vector2 <- c(9,12,15,16,14,11)
>
> # Take these vectors as input to the array.
> result <- array(c(vector1,vector2),dim = c(3,3,2))
> print(result)
, , 1

     [,1] [,2] [,3]
[1,]    6    9   16
[2,]    4   12   14
[3,]    1   15   11

, , 2

     [,1] [,2] [,3]
[1,]    6    9   16
[2,]    4   12   14
[3,]    1   15   11

>
```

Results from the code run above

In arrays, we can provide names to the rows, columns, and matrices. This is done using the **dimnames** parameter.

Shown below is an example of an array with custom names for rows, columns, and matrices.

```r
# Create two vectors of different lengths.
vector1 <- c(50,45,32)
vector2 <- c(15,31,17,13,9,65)
column.names <- c("Batsman1","Batsman2","Batsman3")
row.names <- c("Inning1","Inning2","Inning3")
matrix.names <- c("Matrix_A","Matrix_B")

# Take these vectors as input to the array.
result <- array(c(vector1,vector2),dim = c(3,3,2),
                dimnames = list(row.names,column.names,matrix.names))
print(result)
```

Example of an array

```
, , Matrix_A

        Batsman1 Batsman2 Batsman3
Inning1       50       15       13
Inning2       45       31        9
Inning3       32       17       65

, , Matrix_B

        Batsman1 Batsman2 Batsman3
Inning1       50       15       13
Inning2       45       31        9
Inning3       32       17       65

>
```

Results from the code run above

How to access an element in an array

```r
# Create two vectors of different lengths.
vector1 <- c(50,45,32)
vector2 <- c(15,31,17,13,9,65)
column.names <- c("Batsman1","Batsman2","Batsman3")
row.names <- c("Inning1","Inning2","Inning3")
matrix.names <- c("Matrix_A","Matrix_B")

# Take these vectors as input to the array.
result <- array(c(vector1,vector2),dim = c(3,3,2),
                dimnames = list(row.names,
                                column.names,
                                matrix.names))

# Print the third row of the second matrix of the array.
print(result[3,,2])

# Print the element in the 1st row
# and 3rd column of the 1st matrix.
print(result[1,3,1])

# Print the 2nd Matrix.
print(result[,,2])
```

Example of accessing elements in an array

Elements in an array can be retrieved using the column, row, and matrix index of the element.

Shown below is a code snippet that illustrates how we can access different elements in an array.

```
> # Create two vectors of different lengths.
> vector1 <- c(50,45,32)
> vector2 <- c(15,31,17,13,9,65)
> column.names <- c("Batsman1","Batsman2","Batsman3")
> row.names <- c("Inning1","Inning2","Inning3")
> matrix.names <- c("Matrix_A","Matrix_B")
>
> # Take these vectors as input to the array.
> result <- array(c(vector1,vector2),dim = c(3,3,2),
+                 dimnames = list(row.names,column.names,matrix.names))
>
> # Print the third row of the second matrix of the array.
> print(result[3,,2])
Batsman1 Batsman2 Batsman3
      32       17       65
>
> # Print the element in the 1st row and 3rd column of the 1st matrix.
> print(result[1,3,1])
[1] 13
>
> # Print the 2nd Matrix.
> print(result[,,2])
        Batsman1 Batsman2 Batsman3
Inning1       50       15       13
Inning2       45       31        9
Inning3       32       17       65
>
```

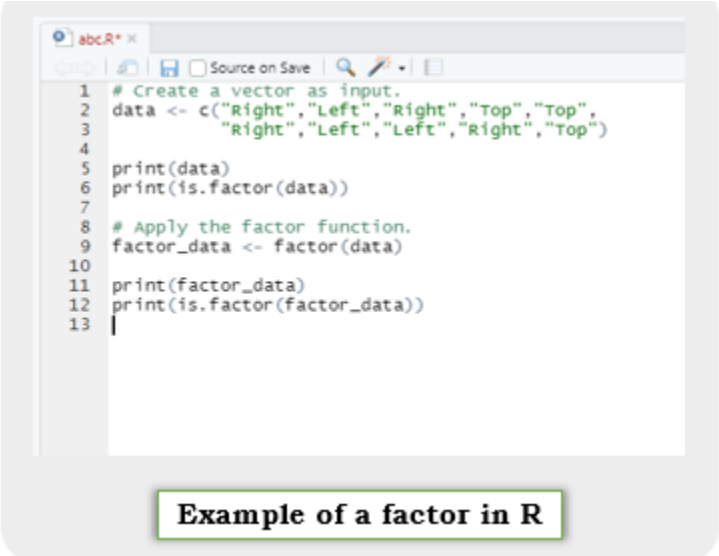
Results from the code run above

Factors in R programming

In R, factors are the data objects used to categorize the data and store it as levels. Factors can store both strings and integers. They are generally used in columns that have a finite number of unique values. Factors are helpful in the data analysis for statistical modeling.

Factors in R are created using the factor() function. The input parameter for this function is a vector.

Shown below is an example of implementing factors in R:



Example of a factor in R

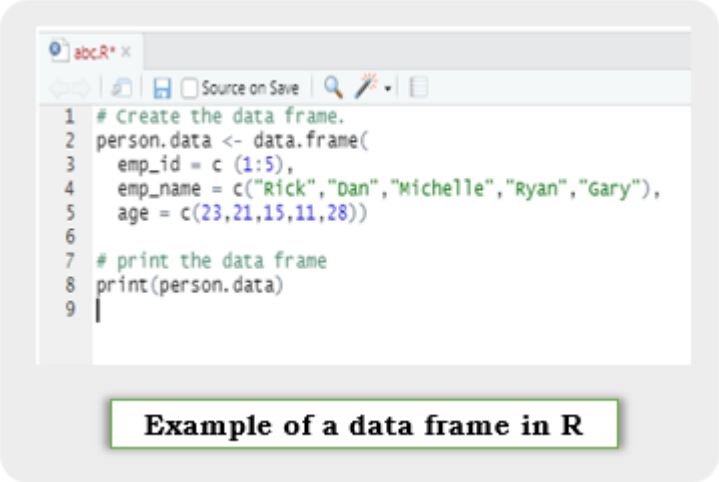
Results from the code run above

Data frames in R programming

In R, a data frame can be defined as a table-like structure used to store data. In a data frame, each column contains the values of each variable. Here, each row contains one set of values related to each column. In a data frame, the column names are non-empty, and the row names should be unique.

Data frames are made up of data that are of numeric, factor, or character data type. Here, each column should contain the same number of data items.

To create a data frame, we use the **data.frame()** function.



Example of a data frame in R

Shown below is an example of a simple data frame:

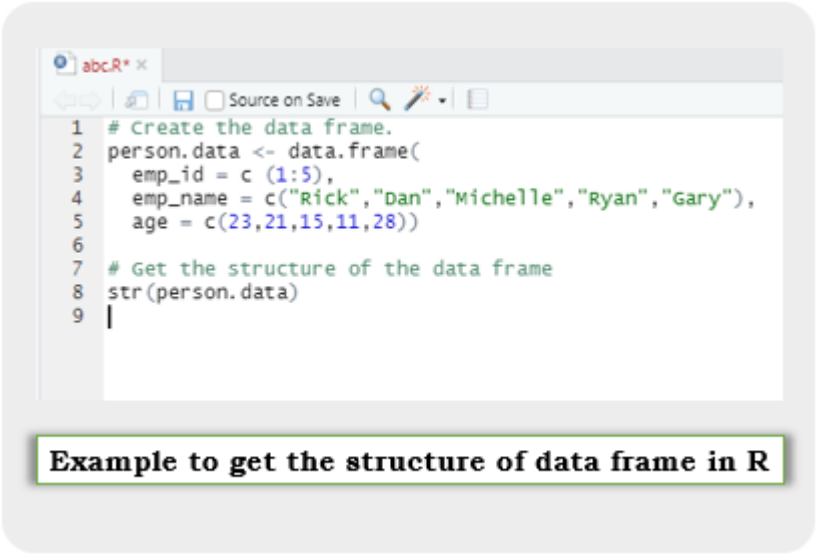
```
  emp_id emp_name age
1      1     Rick  23
2      2      Dan  21
3      3 Michelle  15
4      4     Ryan  11
5      5     Gary  28
```

Results from the code run above

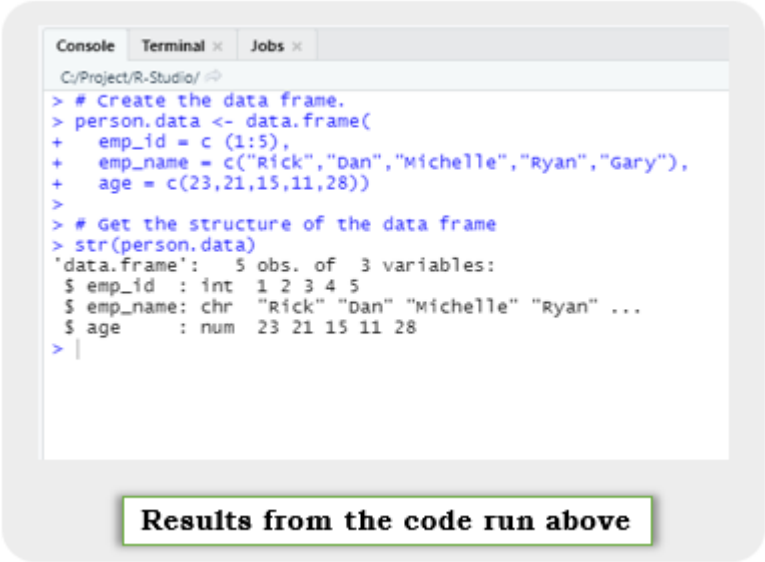
Structure of the data frame

In R, we can get the structure of a data frame using the **str()** function.

Shown below is an example of str() function to get the structure of a data frame.



Example to get the structure of data frame in R



Results from the code run above

Retrieving the summary of data in a data frame

We can get the statistical summary and nature of the data in a data frame by applying the **summary()** function.

Shown below is an example of a **summary()** function to summarize data in a data frame.

```
# Create the data frame.
person.data <- data.frame(
   emp_id = c (1:5),
   emp_name = c("Rick","Dan","Michelle","Ryan","Gary"),
   age = c(23,21,15,11,28))

# Print the summary of the data in the data frame
print(summary(person.data))
```

Example to get the summary of data frame in R

```
> # Create the data frame.
> person.data <- data.frame(
+    emp_id = c (1:5),
+    emp_name = c("Rick","Dan","Michelle","Ryan","Gary"),
+    age = c(23,21,15,11,28))
>
> # Print the summary of the data in the data frame
> print(summary(person.data))
     emp_id     emp_name              age
 Min.   :1   Length:5           Min.   :11.0
 1st Qu.:2   Class :character   1st Qu.:15.0
 Median :3   Mode  :character   Median :21.0
 Mean   :3                      Mean   :19.6
 3rd Qu.:4                      3rd Qu.:23.0
 Max.   :5                      Max.   :28.0
>
```

Results from the code run above

How to extract data from a data frame

We can extract a specific column from a data frame using the column name.

Shown below is an example of extracting data from a data frame.

abc.R*

Source on Save

```
1  # Create the data frame.
2  person.data <- data.frame(
3    emp_id = c (1:5),
4    emp_name = c("Rick","Dan","Michelle","Ryan","Gary"),
5    age = c(23,21,15,11,28))
6
7  # Extract Specific columns.
8  result <- data.frame(person.data$age,person.data$emp_name)
9  print(result)
10
```

Example to extract data from a data frame in R

Console | Terminal | Jobs

C:/Project/R-Studio/

```
> # Create the data frame.
> person.data <- data.frame(
+   emp_id = c (1:5),
+   emp_name = c("Rick","Dan","Michelle","Ryan","Gary"),
+   age = c(23,21,15,11,28))
>
> # Extract Specific columns.
> result <- data.frame(person.data$age,person.data$emp_name)
> print(result)
  person.data.age person.data.emp_name
1              23                 Rick
2              21                  Dan
3              15              Michelle
4              11                 Ryan
5              28                 Gary
>
```

Results from the code run above

3. Coding for Data Science using R-Studio

An essential aspect of data science includes data visualization. We can represent such visualizations as scatter plots, box plots, time series plots, bar charts, histograms, pie charts, etc.

Although we have functions to plot scatter plots, box plots, and time series plots in R, we can also plot them by including a package named **ggplot2**.

ggplot2 is a plotting package that simplifies the creation of complex plots from data in a data frame. This package provides a more programmatic interface to specify what variables to plot, how they should be displayed, and other general visual properties. Thus, one needs to make minimal changes if the underlying data source changes or change the visualization from scatter plot to bar plot.[5]

A few of the essential functions under the ggplot2 package include the **ggplot** function & the **geom** functions.

ggplot graphics are built gradually by adding new elements. This approach makes plotting flexible and customizable.

To build a ggplot, the basic template used for generating different types of plot is:

ggplot(data = <DATA>, mapping = aes(<MAPPINGS>)) + <GEOM_FUNCTION>()

Using the ggplot function, we bind the plot to a data frame. This is done using the data argument.

We define an aesthetic mapping (using the aesthetic (aes) function), by selecting the variables to be plotted and specifying how to present them in the graph, e.g. as x/y positions or characteristics such as size, shape, color, etc.

GEOM_FUNCTION represents the graphical representation of the data in the plot in the form of points, lines, or bars. The most common forms of geom functions are:

- geom_point() [used for scatter plots, dot plots, etc.]
- geom_boxplot() [used for boxplots]
- geom_line() [used for trend lines, time series, etc.]

To add a geom function to the ggplot function, we use a '+' operator.

Scatter plot in R

Let us first see how we can create a scatterplot in R without using any package.
For creating a scatterplot in R, we use the **plot()** function.
The basic syntax for creating a scatterplot is:

plot(x, y, main, xlab, ylab, xlim, ylim, axes)

Following is the description of the parameters used:

- **x** is the data set whose values are the horizontal coordinates.
- **y** is the data set whose values are the vertical coordinates.
- **main** is the tile of the graph.
- **xlab** is the label in the horizontal axis.
- **ylab** is the label in the vertical axis.
- **xlim** is the limits of the values of x used for plotting.
- **ylim** is the limits of the values of y used for plotting.
- **axes** indicate whether both axes should be drawn on the plot.

Shown below is an example of a simple scatter plot drawn using plot() function in R.

In this example, we are using values of two columns **disp** & **hp** from the built in **mtcars** data set in R. We are using the values from these two columns to draw a scatter plot in R.

abc.R* ×

Source on Save

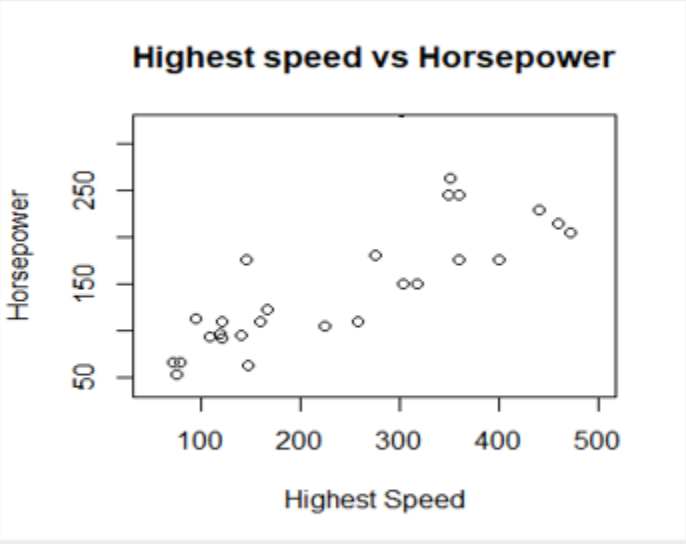
```
1  # Get the input values.
2  input <- mtcars[,c('disp','hp')]
3
4  s <- plot(x = input$disp,y = input$hp,
5       xlab = "Highest Speed",
6       ylab = "Horsepower",
7       xlim = c(50,500),
8       ylim = c(40,320),
9       main = "Highest speed vs Horsepower"
10 )
11 s
12
```

Example to draw scatter plot in R

Shown alongside is the scatterplot drawn in R for the above set of inputs.

The Y-axis displays the **Horsepower** and

The X-axis displays the **Highest speed.**

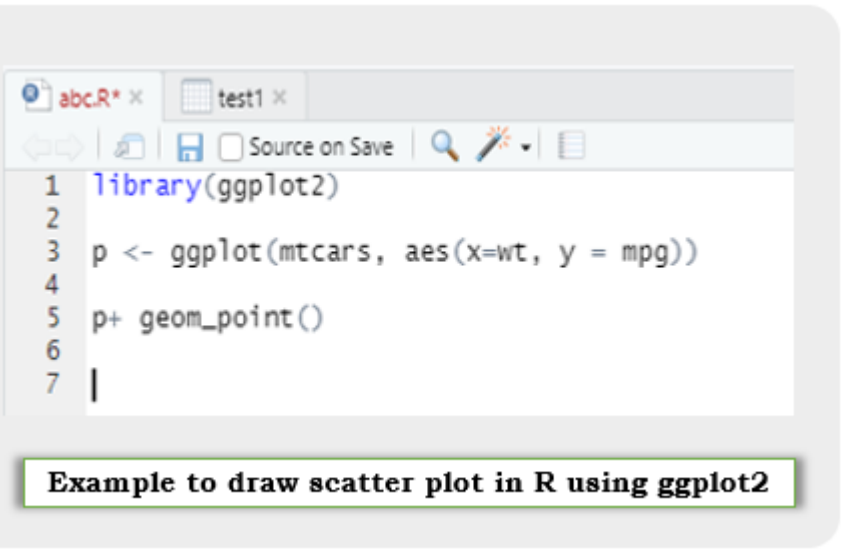


Scatter plot drawn using the above code

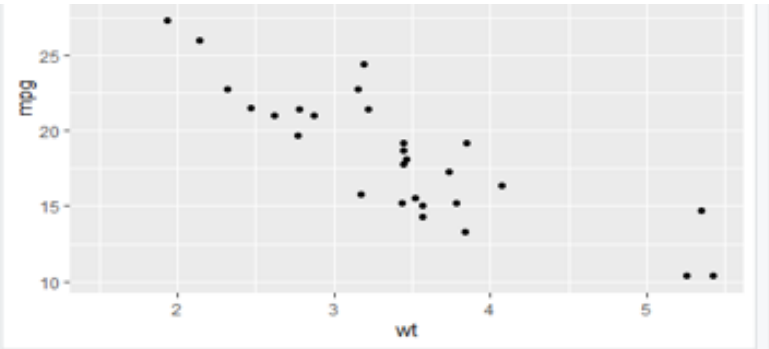
Now let us see how we can plot a scattered plot using the ggplot2 plotting package.

Here, we will see the use of geom_point() along with ggplot())

As stated earlier, in R, we have a predefined dataset named mtcars.



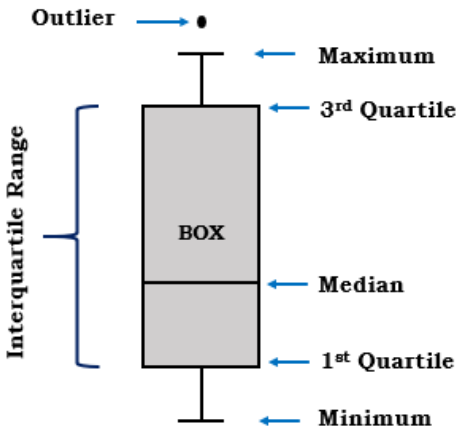
Example to draw scatter plot in R using ggplot2



Scatter plot drawn using the above code

Box plot in R

A box plot is a graphical technique of summarizing a set of data on an interval scale. Boxplots are used extensively in descriptive data analysis. Using this, we can show the shape of the distribution, its central value, and its variability.[6]



A boxplot in R is created using the **boxplot()** function.

The syntax to create a boxplot in R is:

boxplot(x, data, notch, varwidth, names, main)

Following is the description of the parameters used −

- **x** is a vector or a formula.
- **data** is the data frame.
- **notch** is a logical value. Set as TRUE to draw a notch.
- **varwidth** is a logical value. Set as true to draw width of the box proportionate to the sample size.
- **names** are the group labels which will be printed under each boxplot.
- **main** is used to give a title to the graph.

The boxplot() function can also take in formulas of the form Y~X, where Y is a numeric vector grouped according to the value of X.

For demonstrating boxplots in R, we will be using the **airquality** dataset.

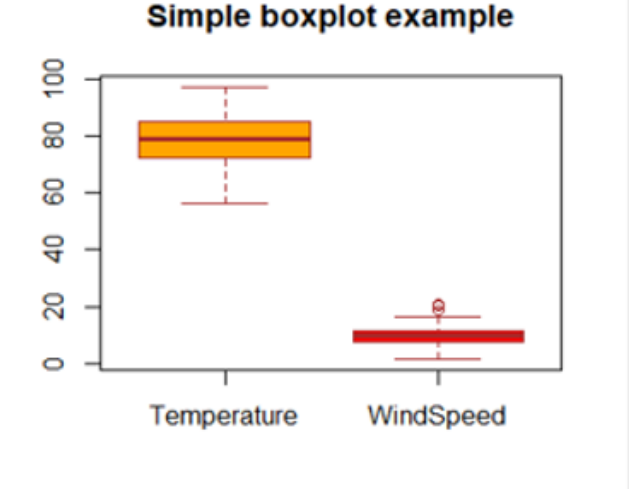
abc.R* ×

Source on Save

```
1  Temperature <- airquality$Temp
2  Wind <- airquality$Wind
3
4  boxplot(Temperature,Wind,
5          main = "Simple boxplot example",
6          names = c("Temperature","WindSpeed"),
7          col = c("orange","red"),
8          border = "brown")
9
10
```

From the airquality data set, we will use the data for the temperature column and windspeed column.

Example to draw box plot in R



Boxplots drawn for temperature and windspeed columns on the airquality data set

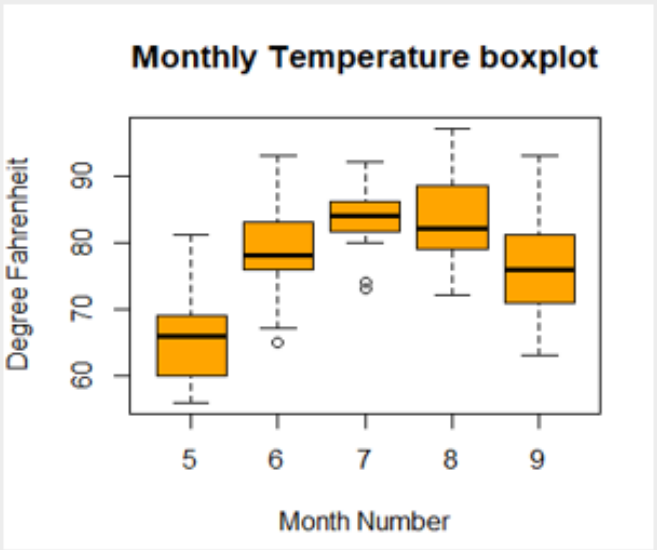
Box plot drawn using the above code

Example of a boxplot where the numeric vector is grouped according to another value.

```
boxplot(Temp~Month,
        airquality,
        main = "Monthly Temperature boxplot",
        xlab = "Month Number",
        ylab = "Degree Fahrenheit",
        col = "orange" )
```

Shown alongside is the example using the airquality dataset. Here the **Temp** is the numeric vector and it has been grouped according to the **Month** number.

Example to draw box plot in R using grouped value



Boxplot to show temperatures grouped according to month number.

The boxplot shown alongside shows that July (month number 7) is relatively hotter than the rest.

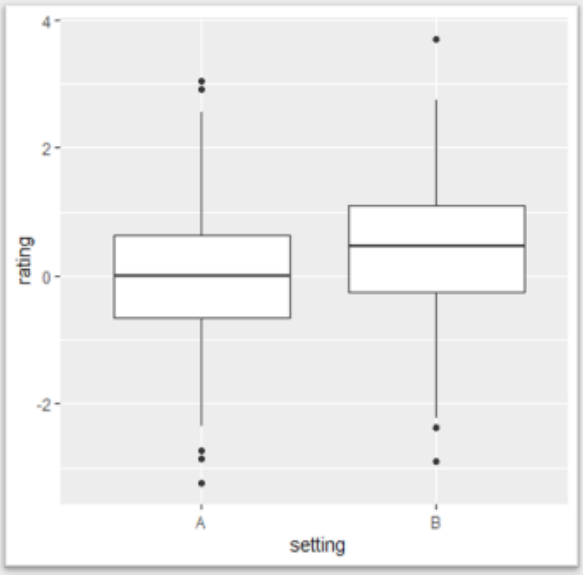
Box plot drawn using the above code

Now let us see how we can plot a boxplot using the ggplot2 plotting package.

Boxplot in R ((use of geom_point() with ggplot())

```
abc.R* ×
Source on Save
1  library(ggplot2)
2  set.seed(1234)
3
4  dat <- data.frame(setting = factor(rep(c("A","B"),each = 300)),
5                    rating = c(rnorm(300), rnorm(300,mean = .5)))
6
7  p <- ggplot(dat, aes(x = setting, y = rating)) + geom_boxplot()
8
9  p
10
```

Example to draw box plot in R using ggplot2



Scatter plot drawn using the above code

Line chart in R

A line chart is a form of a chart created by connecting data points of the data set. Line charts can be used for exploratory data analysis to check the data trends by observing the line graph's line pattern.

To create a line graph in R, we use the plot() function.

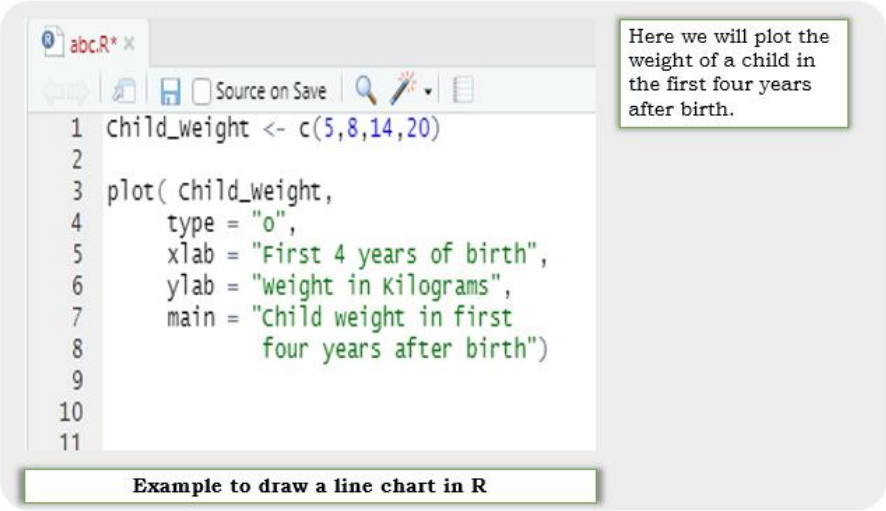
The syntax used to create a line chart in R is:

plot (v, type, xlab, ylab, main, col)

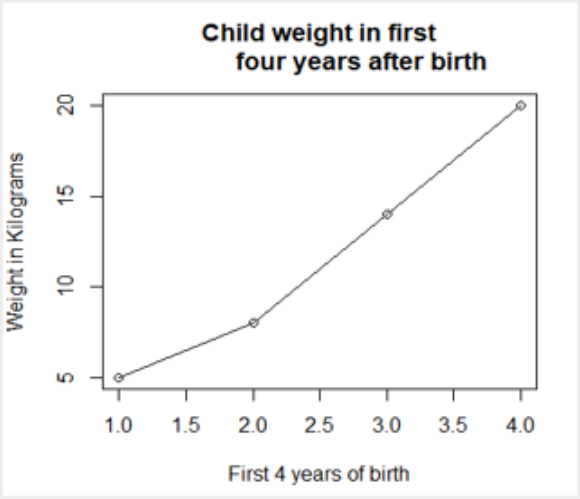
Following is the description of the parameters used –

- **v** is a vector containing the numeric values.
- **type** takes the value "p" to draw only the points, "l" to draw only the lines and "o" to draw both points and lines.
- **xlab** is the label for x axis.
- **ylab** is the label for y axis.
- **main** is the Title of the chart.
- **col** is used to give colors to both the points and lines.

Let us look at an example to draw a line chart in R using plot function



Example to draw a line chart in R



Line chart drawn using the above code

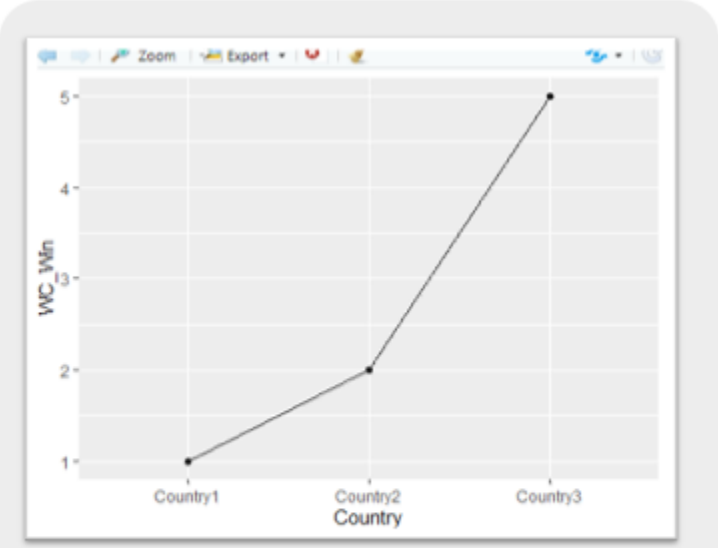
Now let us see how we can draw a line chart using the ggplot2 plotting package.

Line in R (use of geom_line() with ggplot())

```
library(ggplot2)
set.seed(1234)
df <- data.frame(Country = c("Country1","Country2","Country3"),
                 WC_Win = c(1,2,5))

p <- ggplot(data = df,
            aes(x = Country, y = WC_Win, group = 1))+ geom_line() + geom_point()

p
```

Here we are plotting a particular country's world cup win and then connecting them via line chart.

Example to draw a line chart in R using ggplot2



Line chart drawn using the above code

Bar chart in R

The basic syntax for creating a bar chart in R is:

barplot(H,xlab,ylab,main, names.arg,col)

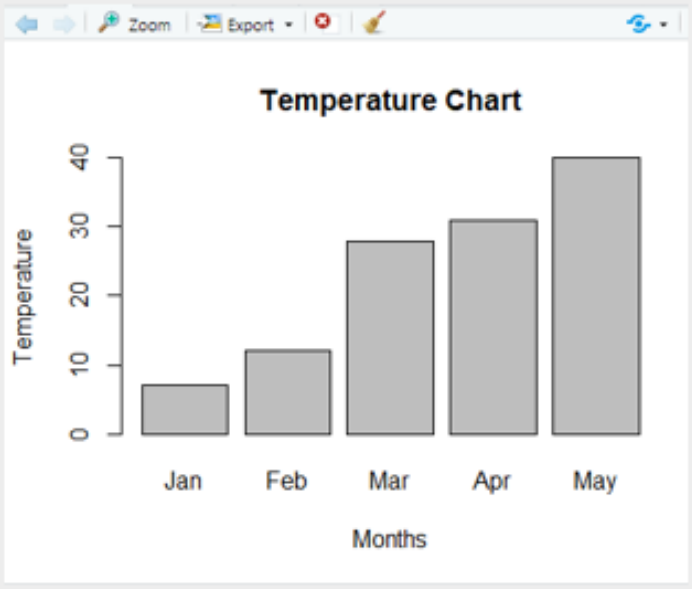
Following is the description of the parameters used:

- **H** is a vector or matrix containing numeric values used in bar chart.
- **xlab** is the label for x axis.
- **ylab** is the label for y axis.
- **main** is the Title of the bar chart.
- **names.arg** is a vector of names appearing under each bar.
- **col** is used to give colors to the bars in the graph.

Given below is an example to draw the bar graph for maximum temperature in Celsius recorded during five consecutive months of a year

```
# Create the data for the chart
Temparature <- c(7,12,28,31,40)
Months <- c("Jan","Feb","Mar","Apr","May")

barplot(Temparature,names.arg = Months,xlab = "Months",
        ylab = "Temperature", main = "Temperature Chart")
```

Example code to draw bar chart in R



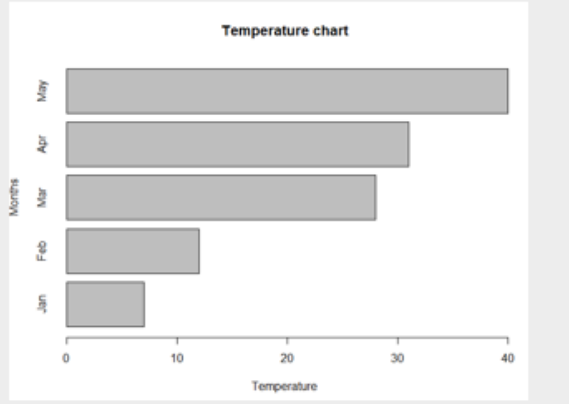
Bar chart drawn using the above code

Bars can also be plotted horizontally by providing the argument horiz = TRUE

```
#Create the data for the chart
Temperature <- c(7,12,28,31,40)
Months <- c("Jan","Feb","Mar","Apr","May")

barplot(Temperature,names.arg = Months,
        ylab= "Months" , xlab = "Temperature",
        main = "Temperature chart", horiz = TRUE)
```

Example code to draw bar chart in R

The horizontal presentation of bar graph for maximum temperature in Celsius was recorded during five consecutive months of a year.



Bar chart drawn using the above code

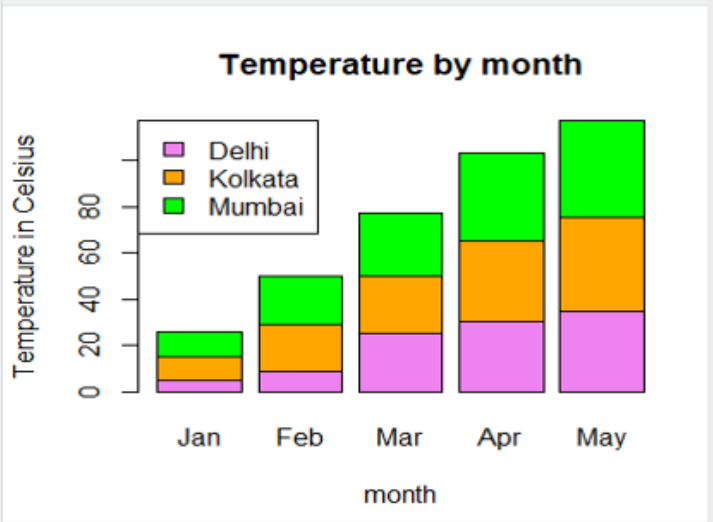
Group bar chart and stacked bar chart

Bar charts can also be created in R with groups of bars and stacks in each bar using a matrix as an input value.

```r
# Create the input vectors.
colors = c("violet","orange","green")
months <- c("Jan","Feb","Mar","Apr","May")
regions <- c("Delhi","Kolkata","Mumbai")

# Create the matrix of the values.
values <- matrix(c(5,9,25,30,35,10,20,25,35,40,11,21,27,38,42),
                 nrow = 3, ncol = 5, byrow = TRUE)

# Create the bar chart
barplot(values, main = "Temperature by month",
        names.arg = months, xlab = "month",
        ylab = "Temperature in Celsius", col = colors)

# Add the legend to the chart
legend("topleft", regions, cex = 1, fill = colors)
```

Here we will represent maximum temperature in Celsius recorded during 5 consecutive months of a year at 3 different cities in India

Example code to draw a stacked bar chart in R

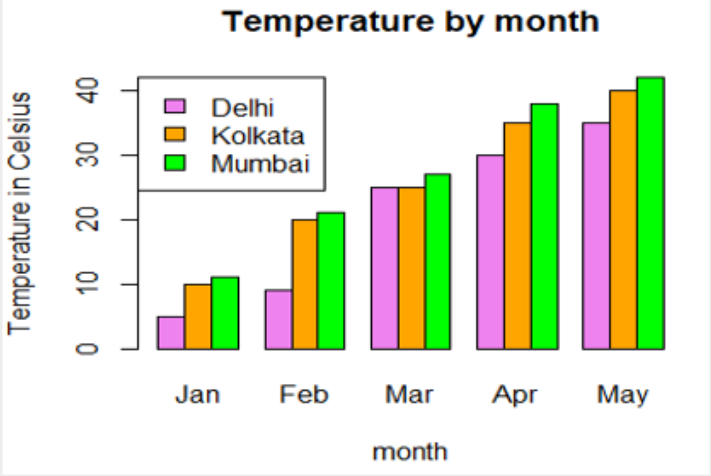


Stacked bar chart drawn using the code written above

Instead of a stacked bar we can have different bars for each element in a column juxtaposed to each other by specifying the parameter **beside = TRUE** in the barplot function as shown below.

```r
# Create the input vectors.
colors = c("violet","orange","green")
months <- c("Jan","Feb","Mar","Apr","May")
regions <- c("Delhi","Kolkata","Mumbai")

# Create the matrix of the values.
Values <- matrix(c(5,9,25,30,35,10,20,25,35,40,11,21,27,38,42),
                 nrow = 3, ncol = 5, byrow = TRUE)

# Create the bar chart
barplot(Values, main = "Temperature by month",
        names.arg = months, xlab = "month",
        ylab = "Temperature in Celsius", col = colors,beside = TRUE)

# Add the legend to the chart
legend("topleft", regions, cex = 1, fill = colors)
```

Example code to draw bar chart in R



Stacked bar chart drawn using the code written above

Histogram in R

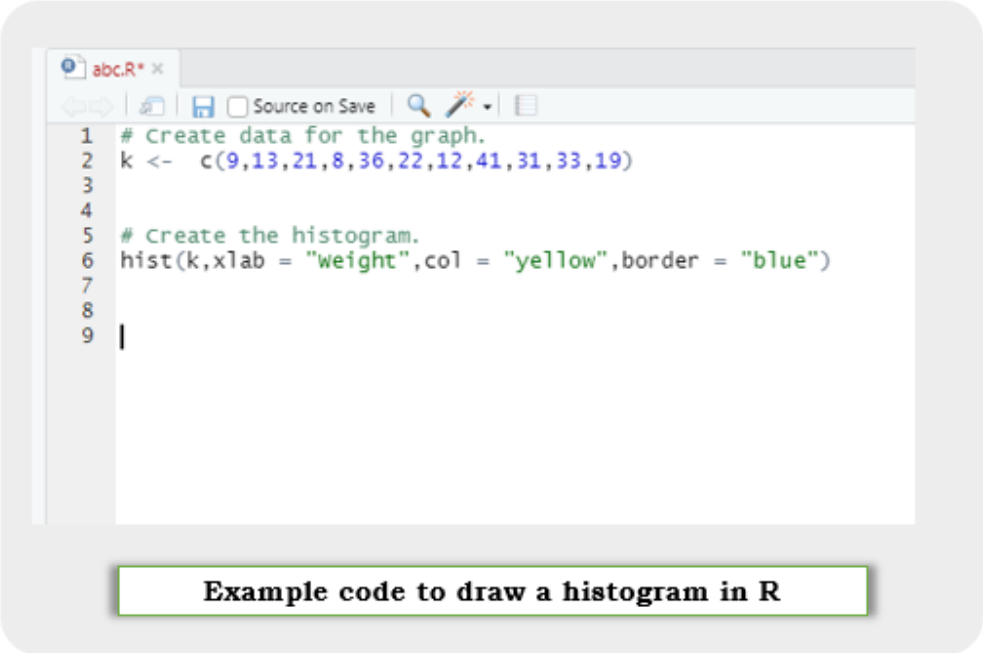
The basic syntax for creating a histogram in R is:

hist(v,main,xlab,xlim,ylim,breaks,col,border)

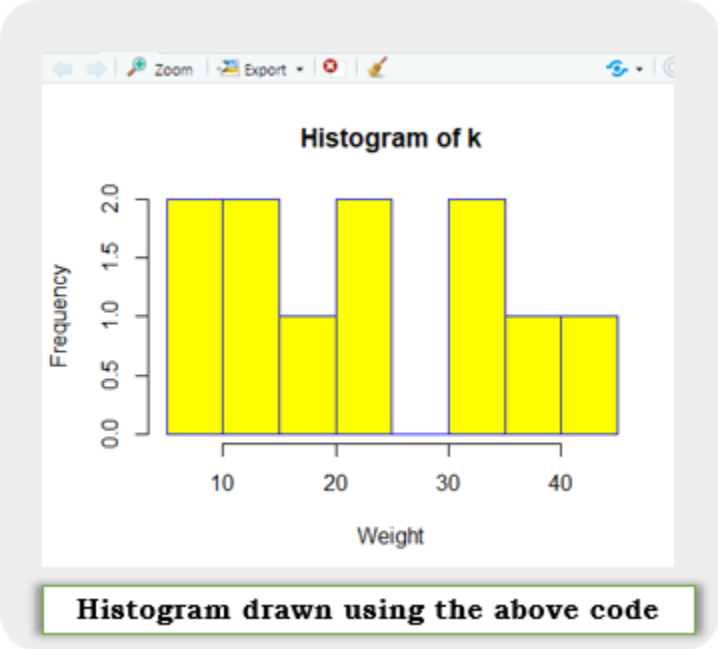
Following is the description of the parameters used:

- **v** is a vector containing numeric values used in histogram.
- **main** indicates Title of the chart.
- **col** is used to set color of the bars.
- **border** is used to set border color of each bar.
- **xlab** is used to give description of x-axis.
- **xlim** is used to specify the range of values on the x-axis.
- **ylim** is used to specify the range of values on the y-axis.
- **breaks** are used to mention the width of each bar.

Shown below is an example to plot a simple histogram using R:



Example code to draw a histogram in R



Histogram drawn using the above code

Pie Chart in R

The basic syntax for creating a pie chart in R is:

pie(x, labels, radius, main, col, clockwise)

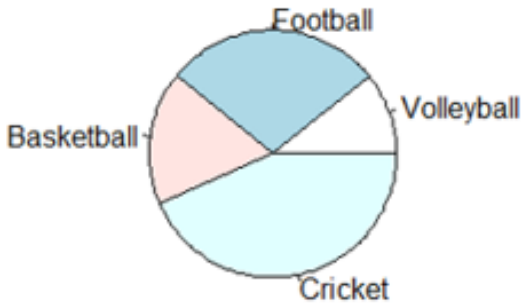
Following is the description of the parameters used:

- **x** is a vector containing the numeric values used in the pie chart.
- **labels** are used to give description to the slices.
- **radius** indicates the radius of the circle of the pie chart.(value between −1 and +1).
- **main** indicates the Title of the chart.
- **col** indicates the color palette.
- **clockwise** is a logical value indicating if the slices are drawn clockwise or anti clockwise.

Shown below is a simple pie chart created using input vector and labels.

```
# Create data for the graph.
y <- c(22, 57, 35, 88)
labels <- c("Volleyball", "Football", "Basketball", "Cricket")


# Plot the chart.
pie(y,labels)
```

Example code to draw a simple pie chart in R



Pie chart drawn using the above code

We can modify the above pie chart with a custom Title and colors

```r
# Create data for the graph.
y <- c(22, 57, 35, 88)
labels <- c("Volleyball", "Football", "Basketball", "Cricket")


# Plot the chart.
pie(y, labels, main = "Sports pie chart", col = rainbow(length(y)))

```

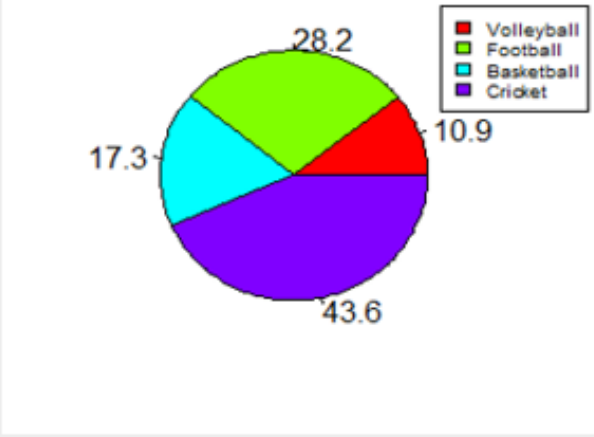
Example code to draw a pie chart in R with custom title and colors

Pie chart drawn using the above code

The above pie chart can also be presented in the form of slice percentages and chart legend

```r
# Create data for the graph.
y <- c(22, 57, 35, 88)
labels1 <- c("Volleyball", "Football", "Basketball", "Cricket")

piepercent<- round(100*y/sum(y), 1)


# Plot the chart.
pie(y, labels = piepercent, main = "Sports pie chart",col = rainbow(length(y)))
legend("topright", labels1, cex = 0.6,
       fill = rainbow(length(y)))
```

Example code to draw a pie chart in R with slice percentage and chart legends



Pie chart drawn using the above code

4. Code examples with R-Studio

In this section, we will determine how to perform statistical analysis using R studio. For this, we will be using many built-in functions. Most of these functions are part of the R base package.

The functions being referred to here are mean, median, and mode.

Mean

We calculate the mean by taking the sum of values and dividing it by the number of values in the data series.

To calculate the mean in R, we use the mean() function. The mean() function in R takes a vector as an input along with the other arguments and generates the result.

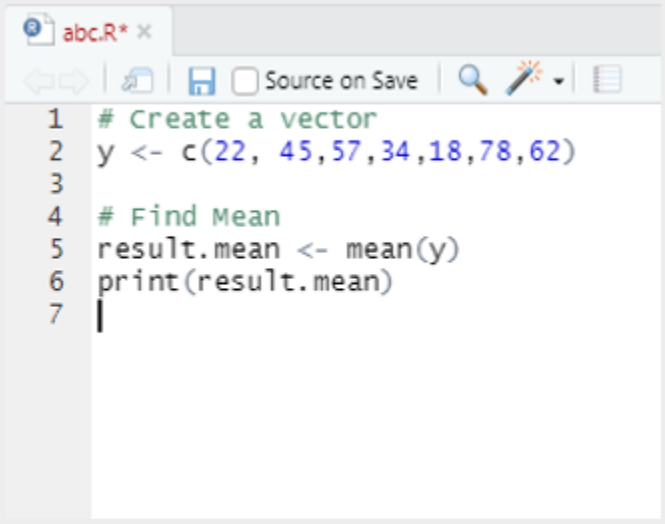
The syntax for calculating mean in R is:

mean(x, trim = 0, na.rm = FALSE, ...)

Following is the description of the parameters used:

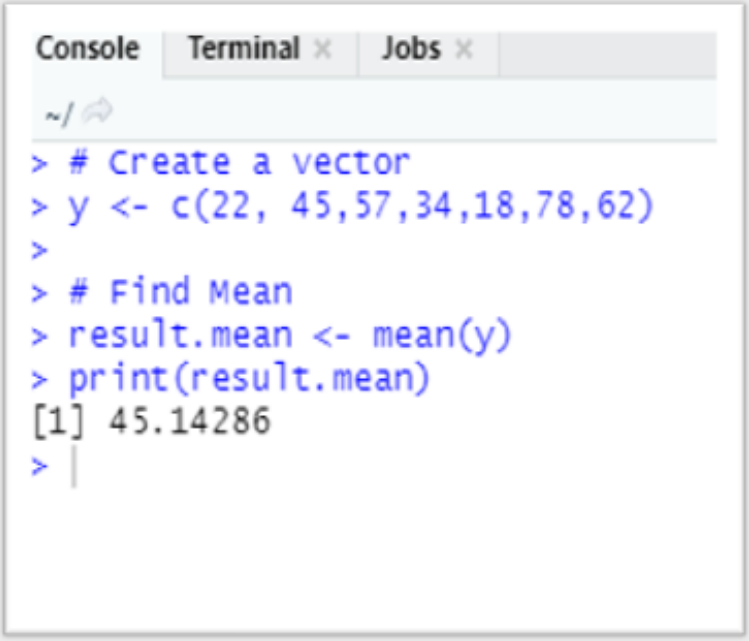
- **x** is the input vector.
- **trim** is used to drop some observations from both end of the sorted vector.
- **na.rm** is used to remove the missing values from the input vector.

Shown below is an example of calculating the mean of a given input vector.



Here we have taken a vector of seven numbers. We then calculate the mean of the seven numbers in the vector by using the mean() function.

Example code to calculate mean in R



As shown in the result, the mean for the given numeric vector is coming as 45.14286.

Result generated for the above executed code

Use of **trim** parameter in the mean() function:

When the trim parameter is used in the function, the vector's values get sorted. Then the required number of observations are dropped from both ends from the calculation of the mean.

So, when trim = 0.2, 2 values from both the right end and left end are dropped from the mean calculation.

Shown below is an example of calculating the mean of a given input vector, having a trim parameter

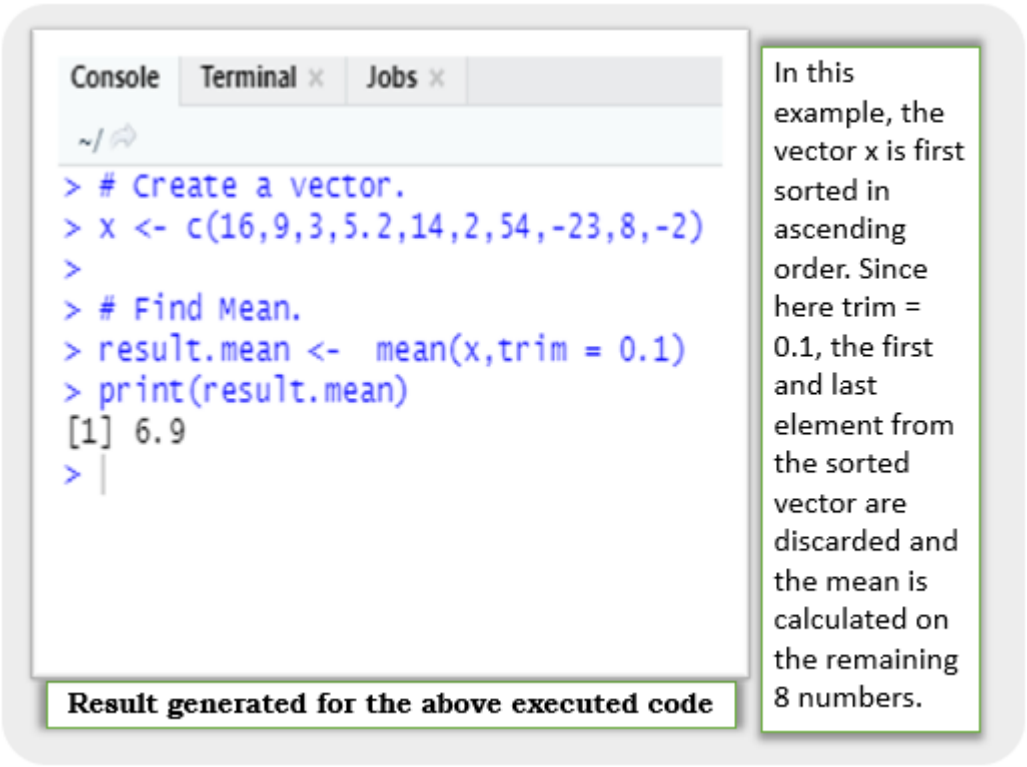
abc.R*

Source on Save

```
1  # Create a vector.
2  x <- c(16,9,3,5.2,14,2,54,-23,8,-2)
3
4  # Find Mean.
5  result.mean <-  mean(x,trim = 0.1)
6  print(result.mean)
7
8
9
```

In this example, we have considered a vector of length 10 and calculated the mean with a trim value of 0.1

Example code to calculate mean with trim value in R



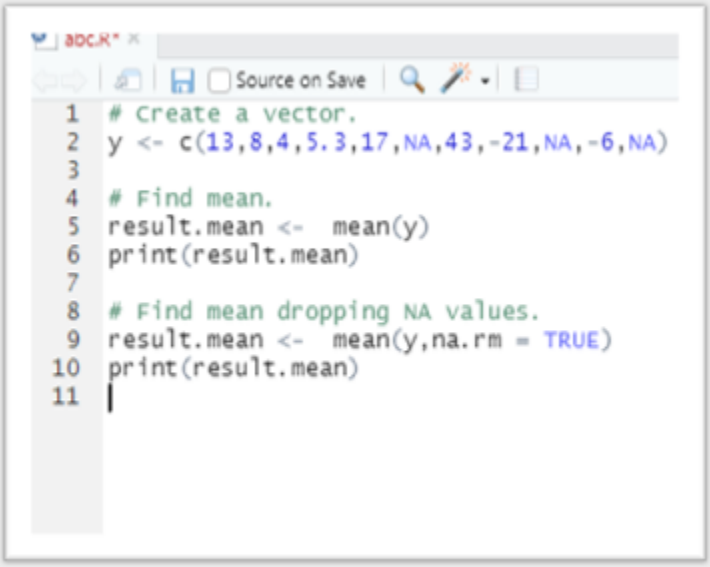
Result generated for the above executed code

Use of **na.rm** parameter in the mean() function:

In some situations, one or more value may be NA. Suppose the vector contains a missing value (populated in the vector as NA). In that case, the mean function by default returns NA.

To drop the missing value in a vector, we set the na.rm parameter to TRUE (i.e., na.rm = TRUE), which means remove the NA values.

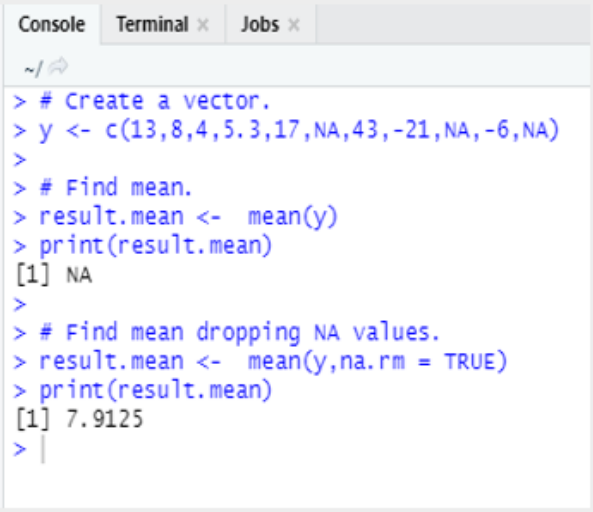
Shown below is an example of calculating the mean of a given input vector with missing values:



```
# Create a vector.
y <- c(13,8,4,5.3,17,NA,43,-21,NA,-6,NA)

# Find mean.
result.mean <-  mean(y)
print(result.mean)

# Find mean dropping NA values.
result.mean <-  mean(y,na.rm = TRUE)
print(result.mean)
```

In this example, we have considered a vector having 3 missing value.

Example code to calculate mean of a vector having null values



```
> # Create a vector.
> y <- c(13,8,4,5.3,17,NA,43,-21,NA,-6,NA)
>
> # Find mean.
> result.mean <-  mean(y)
> print(result.mean)
[1] NA
>
> # Find mean dropping NA values.
> result.mean <-  mean(y,na.rm = TRUE)
> print(result.mean)
[1] 7.9125
>
```

In the calculation of mean for the vector, if we don't declare na.rm = TRUE, in that case result for the mean calculation returns NA.

In the calculation of mean for the vector, if we declare na.rm = TRUE, in that case the missing values are ignored and mean gets calculated on the rest of the elements in the vector.

Result generated for the above executed code

Median

The median can be defined as the middlemost value in a data series.

To calculate the median value for a data series in R, we use the median() function.

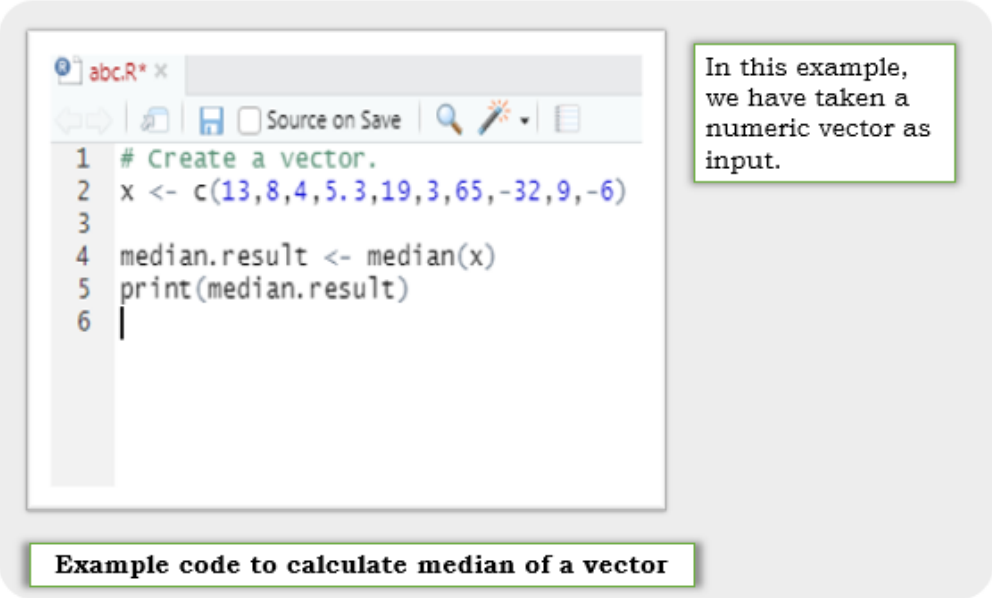
The syntax for calculating median in R is:

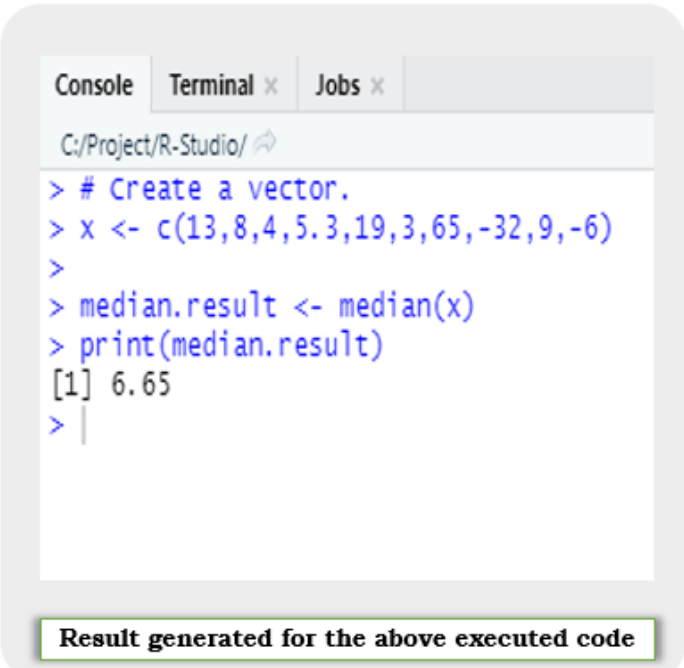
median(x, na.rm = FALSE)

Following is the description of the parameters used –

- **x** is the input vector.
- **na.rm** is used to remove the missing values from the input vector.

Shown below is an example of calculating the median of a numeric vector



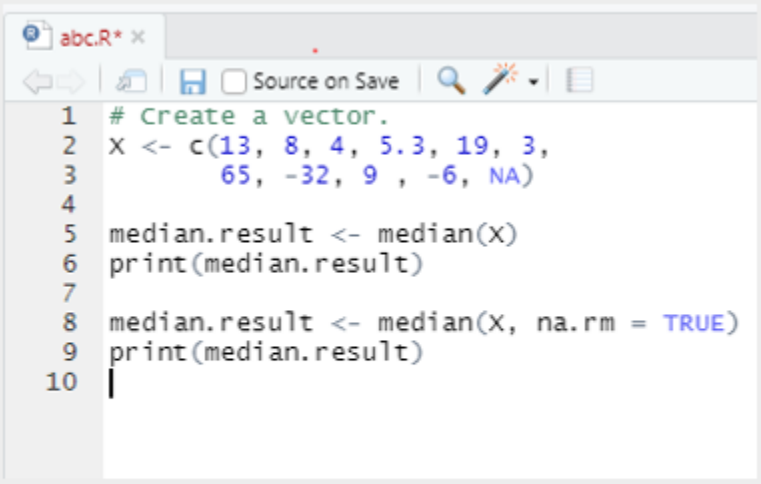


Use of **na.rm** parameter in the median() function:

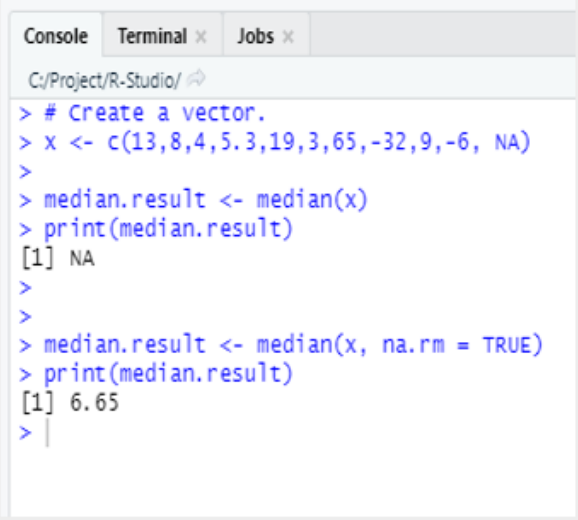
In some situations, one or more value may be NA. Suppose the vector contains a missing value (populated in the vector as NA). In that case, the median function by default returns NA.

To drop the missing value in a vector, we set the na.rm parameter to TRUE (i.e., na.rm = TRUE), which means remove the NA values.

Shown below is an example of calculating the median of a given input vector with a missing value



Example code to calculate median of a vector having NULL values



In the calculation of median for the vector, if we don't declare na.rm = TRUE, in that case result for the median calculation returns NA.

In the calculation of median for the vector, if we declare na.rm = TRUE, in that case the missing values are ignored and median gets calculated on the rest of the elements in the vector.

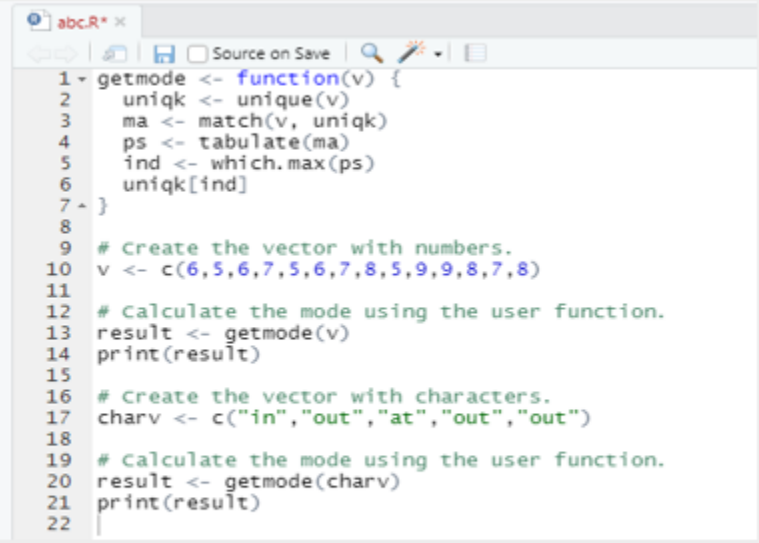
Result generated for the above executed code

Mode

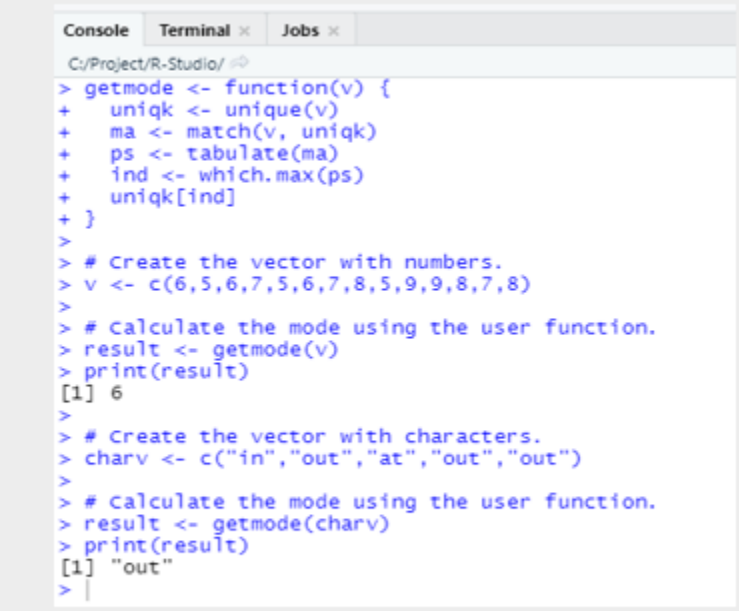
A mode is defined as the value with the highest frequency of occurrences in a set of data. The mode can be determined both for numeric and character data.

For calculating the mode, we do not have any inbuilt function in R. Here; we create a user-defined function to get the mode of a data set.

Shown below is an example of a user-defined function that takes a vector as input and returns the mode value as output.



Example code to calculate mode of a vector



Result generated for the above executed code

Recap

- In this chapter, we have learnt about use of R Studio for programming in R.
- We have learnt about the different data objects in R.
- We have learnt about methods to generate different statistical visualization using R.
- We have learnt the ways to calculate mean, median & mode using R as the programming language.

Exercises

Objective Type Questions

1) R is an _____ programming language?
   a) Closed source
   b) GPL
   c) Open source
   d) Definite source

2) How many types of R objects are present in R data type?
   a) 4
   b) 5
   c) 6
   d) 7

3) In vector, where all arguments are forced to a common type, please tick the correct order:
   a) Expression <logical < double
   b) Double < logical <NULL
   c) Integer < character < expression
   d) Character < logical < raw

4) Which function is used to create the vector with more than one element?
   a) Library ()
   b) plot ()
   c) c ()
   d) par ()

5) Which of the following is incorrect regarding List in R programming?
   a) A list is a type of R object containing elements of different types like numbers, strings, vectors.
   b) A list can also contain another list within it
   c) Elements in a list can be accessed using the index of the element in the list.

d) The addition or deletion of the elements can only be done at the middle of the list.

6) A _____ is a two-dimensional rectangular data set.
   e) Vector
   f) Lists
   g) Matrix
   h) Functions

7) Which function takes a dim attribute which creates the required number of dimensions?
   i) Vector
   j) Array
   k) Matrix
   l) Lists

8) By what function we can create data frames?
   m) Data.frames ()
   n) Data.sets ()
   o) Function ()
   p) C ()

9) Which are indexed by either row or column using a specific name or number?
   q) Datasets
   r) Data frames
   s) Data
   t) Functions

10) The geom function used to plot scatter plot while using ggplot2 package is:
   u) geom_point()
   v) geom_boxplot()
   w) geom_line()

x) None of the above

Standard Questions

1) Write a program in R to create a multi-element vector having the first ten natural numbers and print it.
2) Suppose we have two vectors, V1 = [2, 3, 0, 5] and V2 = [6, 1, 7, 0]. What will be the result of the following?
   y) V1+V2
   z) V1- V2
   aa) V1*V3
   bb) V1/V2
3) Write an R program to create two numeric matrices and perform arithmetic operations like addition, subtraction, multiplication, and division.
4) Draw a bar plot in R, showing the number of days (y-axis) in each month for the year 2016(x-axis).
5) The table below shows the number of water filters sold in the first six months of the year 2019 by two different stores, Store A & Store B.

Month	Jan	Feb	Mar	Apr	May	Jun
Store A	32	34	35	28	37	45
Store B	28	22	34	36	25	20

   Use the data shown in the table above to draw a side by side stacked chart, having Month in X axis, Number of water filters sold in Y axis and legend for bars of Store A & Store B.
6) Draw a pie chart in R to represent days in each month for the year 2016.
7) For a given vector (7, 3, 8, 6, 5, 7, 1, 9, 2, 7, 6), write a program in R to calculate and display the mean, median, and mode of the vector.

Higher Order Thinking Skills (HOTS)

1) In R, we have an inbuilt dataset named **Orange**. Plot a **scatter plot** showing age(x-axis) vs circumference(y-axis). The scatter plot should be plotted first using the normal function in R and then using the ggplot2 library.
2) In R, we have an inbuilt dataset named **Orange**. Plot a boxplot showing age(x-axis) vs circumference(y-axis). The boxplot should be plotted first using the normal function in R and then using the ggplot2 library.
3) In R, we have an inbuilt data set named **Orange**. Plot a histogram for the circumference of the Orange dataset.
4) In R, we have an inbuilt data set named **BOD**. Use the data set to draw a line chart plotting Time vs. Demand.

Applied Project

There is an inbuilt data set named iris present in R. Use the iris dataset to plot the box plot of Sepal length for different species present. Also plot a scattered plot for Sepal width against Sepal length.

References

Bargagliotti, A., Franklin, C., Arnold, P. and Gould, R., 2020. Pre-K-12 Guidelines for Assessment and Instruction in Statistics Education II (GAISE II). American Statistical Association

Creative Maths. 2021. Creative Maths: A world of mathematicians. [ONLINE] Available at: https://creativemaths.net/videos/video-confidence-intervals. [Accessed 25 February 2021]

Data Carpentry contributors. 2021. Data visualization with ggplot2. [ONLINE] Available at: https://datacarpentry.org/R-ecology-lesson/04-visualization-ggplot2.html. [Accessed 25 February 2021]

Deborah J. Rumsey. 2021. What a Boxplot Can Tell You about a Statistical Data Set - dummies. [ONLINE] Available at: https://www.dummies.com/education/math/statistics/what-a-boxplot-can-tell-you-about-a-statistical-data-set. [Accessed 25 February 2021]