SIMPLE FILTERING AND SORTING USING R

A PROJECT REPORT

Submitted by

ROSHITH S (2303811724322091)

in partial fulfillment of requirements for the award of the course

AGI1252 - FUNDAMENTALS OF DATA SCIENCE USING R

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

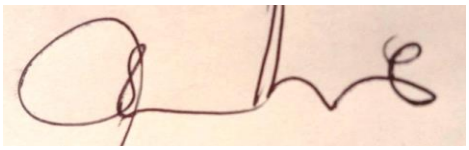
JUNE- 2025

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"SIMPLE FILTERING AND SORTING USING R"** is the bonafide work of **ROSHITH S (2303811724322091)** who carried out the project work during the academic year 2024 - 2025 under my supervision.



SIGNATURE

Dr.T. AVUDAIAPPAN, M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of Artificial Intelligence

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

SIGNATURE

Ms.S.Murugavalli., M.E.,(Ph.D).,

SUPERVISOR

ASSISTANT PROFESSOR

Department of Artificial Intelligence

K.Ramakrishnan College of Technology (Autonomous)

Samayapuram–621112.

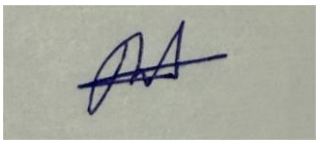
Submitted for the viva-voce examination held on ……**02.06.2025**.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

I declare that the project report on **“SIMPLE FILTERING AND SORTING USING R”** is the result of original work done by me and best of my knowledge, similar work has not been submitted to **“ANNA UNIVERSITY CHENNAI”** for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **AGI1252 - FUNDAMENTALS OF DATA SCIENCE USING R**

.



Signature

ROSHITH S

Place: Samayapuram

Date:30.05.2025

ACKNOWLEDGEMENT

It is with great pride that I express my gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express my sincere thanks to my beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. T. AVUDAIAPPAN, M.E.,Ph.D.,** Head of the department, **ARTIFICIAL INTELLIGENCE** for providing his encourage pursuing this project.

I express my deep expression and sincere gratitude to our project supervisor **Ms.S.Murugavalli., M.E.,(Ph.D).,** Department of **ARTIFICIAL INTELLIGENCE,** for her incalculable suggestions, creativity, assistance and patience which motivated me to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of my departments who rendered their help during the period of the work progress.

INSTITUTE

Vision:

- To serve the society by offering top-notch technical education on par with global standards.

Mission:

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of industry and society.
- Be an institute with world class research facilities.
- Be an institute nurturing talent and enhancing competency of students to transform them as all – round personalities respecting moral and ethical values.

DEPARTMENT

Vision:

- To excel in education, innovation, and research in Artificial Intelligence and Data Science to fulfil industrial demands and societal expectations.

Mission

- To educate future engineers with solid fundamentals, continually improving teaching methods using modern tools.
- To collaborate with industry and offer top-notch facilities in a conducive learning environment.
- To foster skilled engineers and ethical innovation in AI and Data Science for global recognition and impactful research.
- To tackle the societal challenge of producing capable professionals by instilling employability skills and human values.

PROGRAM EDUCATIONAL OBJECTIVES (PEO)

- **PEO1:** Compete on a global scale for a professional career in Artificial Intelligence and Data Science.
- **PEO2:** Provide industry-specific solutions for the society with effective communication and ethics.
- **PEO3** Enhance their professional skills through research and lifelong learning initiatives.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1:** Capable of finding the important factors in large datasets, simplify the data, and improve predictive model accuracy.
- **PSO2:** Capable of analyzing and providing a solution to a given real-world problem by designing an effective program.

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply knowledge of mathematics, natural science, computing, engineering fundamentals, and an engineering specialization to develop solutions to complex engineering problems.

2. **Problem analysis:** Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development.

3. **Design/development of solutions:** Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required.

4. **Conduct investigations of complex problems:** Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions.

5. **Engineering Tool Usage:** Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems.

6. **The Engineer and The World:** Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment.

7. **Ethics:** Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws.

8. **Individual and Collaborative Team work:** Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.

9. **Communication:** Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences.

10. **Project management and finance:** Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.

11. **Life-long learning:** Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change.

ABSTRACT

Simple filtering and sorting in R are essential operations that form the backbone of data preprocessing and analysis, allowing users to refine, extract, and organize data according to specific analytical needs. **Filtering** refers to the process of narrowing down a dataset by selecting rows (observations) that satisfy one or more conditions or logical expressions. In base R, this is typically achieved using logical indexing—where conditions like data[data$Age > 30,] select only rows where the Age column is greater than 30—or by using the subset() function, which offers more readable syntax and serves the same purpose. For more intuitive and readable data manipulation, especially with larger datasets, the **dplyr** package provides the filter() function, which is part of the powerful **tidyverse** ecosystem. This function allows analysts to chain operations using the pipe operator (%>%) and supports multiple filtering criteria with logical operators such as & (AND), | (OR), and ! (NOT), making it highly expressive and user-friendly. **Sorting**, on the other hand, involves arranging the rows of a dataset in a specific sequence based on one or more columns. This can help in identifying trends, detecting outliers, or preparing data for reporting and visualization. In base R, the order() function is used for sorting; for example, data[order(data$Salary),] sorts the dataset in ascending order by salary, and data[order(-data$Salary),] sorts it in descending order. The **dplyr** package complements this with the arrange() function, which simplifies the syntax and supports the use of desc() for descending order. Multiple columns can be sorted simultaneously to resolve ties or prioritize sorting criteria—for instance, arranging data first by department and then by age. Combining filtering and sorting is a common practice, where datasets are first filtered to isolate relevant observations and then sorted to bring order and meaning to the results, often done seamlessly in a single pipeline using dplyr's expressive syntax.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD DATA SCIENCE USING R PROGRAMMING FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
Data manipulation is a fundamental step in data analysis, and R provides powerful tools for filtering and sorting data efficiently. Simple filtering in R involves selecting rows based on specific conditions using logical operators and functions like subset(), filter() from the *dplyr* package, or direct indexing with logical expressions. Sorting, on the other hand, helps organize data in ascending or descending order using functions such as order() or arrange() from *dplyr*. These operations are essential for refining large datasets to focus on relevant observations, identify trends, and prepare data for further statistical analysis or visualization. This abstract explores the basic techniques of filtering and sorting in R, emphasizing their role in making datasets more manageable and insights more accessible.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3**	**PSO1 -3** **PSO2 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.1 Objective

The objective of implementing simple filtering and sorting in R is to enable efficient extraction, organization, and refinement of data to support accurate and meaningful analysis. These techniques allow users to isolate specific subsets of data based on defined conditions and arrange them in a logical order to identify trends, patterns, and outliers. By mastering filtering and sorting, analysts can streamline the data preparation process, enhance clarity in datasets, and lay a solid foundation for more complex statistical operations, modeling, and visualization tasks in the R programming environment.

1.2 Overview

Simple filtering and sorting in R are crucial components of data manipulation that form the foundation for effective data analysis, especially when working with large or complex datasets. These operations are designed to make datasets more manageable, relevant, and easier to interpret by selectively extracting and organizing data based on specific criteria. **Filtering** allows analysts to isolate rows of data that meet certain logical conditions, such as selecting records where values exceed a threshold, belong to a certain category, or satisfy multiple combined criteria using logical operators like AND (&), OR (|), and NOT (!). This helps in narrowing down large datasets to focus only on the information of interest. In base R, filtering is typically done using logical indexing or the subset() function, while in more advanced workflows, the filter() function from the **dplyr** package is widely used for its clarity and ease of chaining multiple operations using the pipe operator (%>%). **Sorting**, on the other hand, is the process of arranging rows in a specific order based on one or more columns. This is particularly useful when analyzing trends, ranking data,

organizing results for reports, or preparing datasets for visualization. In base R, the order() function enables sorting in ascending or descending order, whereas in dplyr, the arrange() function serves the same purpose in a more readable format and supports multi-level sorting with the desc() function for descending order. These operations can be seamlessly combined in a single workflow to first filter out irrelevant data and then sort the remaining data for clearer insights. For instance, an analyst might filter for sales data from a specific region and then sort the results by revenue to identify top-performing products. Overall, mastering simple filtering and sorting in R is essential for data cleaning, exploration, and reporting, and provides the groundwork for more advanced analytical techniques. These functions not only enhance efficiency and accuracy but also contribute significantly to the overall quality and clarity of data-driven decision-making processes.

1.3 R Programming Concepts Used

In simple filtering and sorting using R, a wide range of foundational programming concepts come into play, making these operations essential for efficient data manipulation and preparation. At the core of these tasks is the **data frame**, R's primary data structure for handling tabular data, which allows for the storage and organization of variables in rows and columns. **Logical operators** such as >, <, ==, !=, >=, and <= are used in combination with **logical connectors** like & (AND), | (OR), and ! (NOT) to build conditions for filtering rows that meet specific criteria. This process, known as **subsetting**, is done either using square bracket indexing (e.g., data[data$Age > 25,]) or by applying the more readable subset() function in base R. Filtering becomes even more streamlined with the use of the filter() function from the **dplyr** package, which is part of the **tidyverse**, a collection of R packages designed for data science. Similarly, **sorting** data is carried out using the order() function in base R or the

arrange() function in dplyr, both of which allow users to sort data in ascending or descending order, with the desc() function enabling reverse sorting. A critical and widely used concept in dplyr is the **pipe operator (**%>%**)**, which facilitates readable, step-by-step data workflows by passing the result of one function into the next. In addition, understanding how to **reference variables** or columns is essential—either through the $ notation (e.g., data$Age) in base R or directly by name within dplyr functions (e.g., filter(data, Age > 25)). These tasks also rely on **function calling**, **package loading** (e.g., library(dplyr)), and working within the principles of **tidy data**, where each column is a variable and each row is an observation. Together, these R programming concepts provide users with powerful tools to clean, filter, sort, and organize data efficiently, serving as a foundation for more advanced tasks such as data visualization, modeling, and statistical analysis. Mastering these concepts not only improves coding skills in R but also enhances the overall ability to derive meaningful insights from raw datasets.

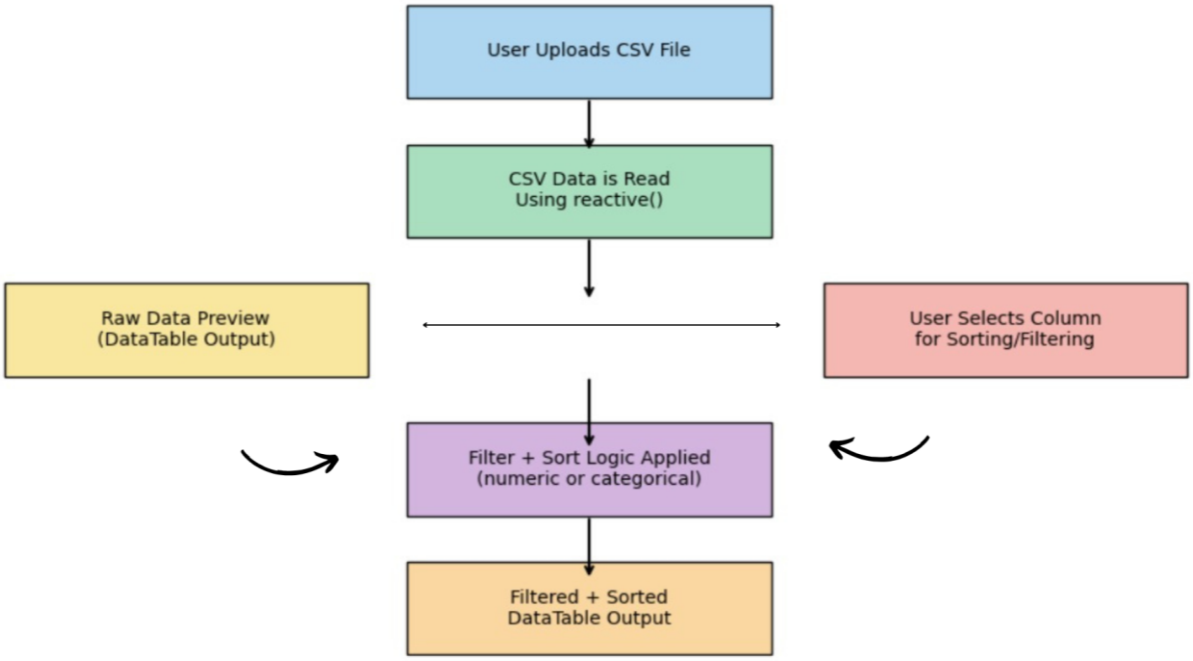
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work involves implementing and demonstrating simple filtering and sorting techniques in R using both base R functions and the dplyr package to efficiently manipulate and analyze structured datasets. The goal is to apply logical conditions to extract specific subsets of data that meet defined criteria (e.g., selecting rows where a numeric value exceeds a threshold or a categorical value matches a given label), and to organize the data in ascending or descending order based on one or more variables. The process will begin with importing or creating a sample data frame, followed by filtering rows using base R methods such as logical indexing and subset(), and then using filter() from the dplyr package for more readable and flexible syntax. Sorting will be demonstrated using the order() function in base R and the arrange() function in dplyr, including multi-level sorting and descending order with desc(). These operations will be combined in a sequential workflow using the pipe operator %>% to illustrate how complex data manipulation tasks can be simplified and streamlined. The proposed work will also emphasize the comparison between base R and dplyr approaches, highlighting the advantages of tidyverse tools in terms of clarity, efficiency, and ease of use. The overall objective is to provide a clear, practical guide to performing essential data filtering and sorting tasks in R, which are critical steps in preparing data for deeper analysis, visualization, and interpretation.

2.2 Block Diagram


User Uploads CSV File
CSV Data is Read
Using reactive()
Raw Data Preview
(DataTable Output)
User Selects Column
for Sorting/Filtering
Filter + Sort Logic Applied
(numeric or categorical)
Filtered + Sorted
DataTable Output

CHAPTER 3

MODULE DESCRIPTION

3.1 File Uplode Module

This module enables users to upload a CSV file from their local system into the Shiny application. It uses the fileInput() UI element and a reactive function (dataset()) to read and store the contents of the uploaded file. This forms the foundation for all subsequent filtering and sorting operations, as it dynamically loads the user's dataset into the app.

3.2 Dynamic Filtering UI Module

This module dynamically generates a user interface for filtering data. Once the dataset is uploaded, it populates a dropdown menu (selectInput) with the column names, allowing the user to choose which column to filter by. It also generates a text input box where the user can enter a search term or keyword. The filter is applied using grepl() to perform case-insensitive, partial string matching within the selected column.

3.3 Dynamic Sorting UI Module

This module provides users with the ability to sort the dataset based on a selected column and direction. It creates a dropdown (selectInput) for selecting the sort column and a radio button group (radioButtons) for choosing between ascending or descending order. Sorting is executed using the arrange() and desc() functions from the dplyr package, depending on the selected direction.

3.4 Filtered and Sorted Data Processing Module

This is the core processing module that performs the actual filtering and sorting logic. It uses a reactive expression (filtered_data()) to monitor inputs

such as the selected filter column, filter value, sort column, and sort direction. Based on these inputs, it filters the dataset using grepl() and sorts it using dplyr functions, returning the final processed data to be rendered and downloaded.

3.5 Data Table Display and Download Module

This module is responsible for presenting the filtered and sorted dataset in an interactive table format using DT::renderDT. It also provides a download feature (downloadButton and downloadHandler) that allows users to export the resulting dataset as a CSV file. This enhances usability by offering immediate visibility and accessibility of the refined data.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

Conclusion

The developed Shiny application successfully demonstrates the practical implementation of simple filtering and sorting techniques in R, providing users with an interactive and user-friendly interface for data manipulation. By combining the functionality of base R and the powerful dplyr and DT packages, the app allows users to upload a dataset, dynamically filter specific records based on column values, sort the data in ascending or descending order, and view the results in a responsive data table. Additionally, the ability to download the processed dataset enhances its usability in real-world analytical workflows. This project not only simplifies the process of exploring and organizing datasets but also highlights the power and flexibility of R Shiny for building dynamic, data-driven web applications. Overall, it serves as a foundational tool for users looking to clean, prepare, and understand their data efficiently without needing advanced programming knowledge.

Future Scope

Future Scope:

The current Shiny application provides a solid foundation for basic filtering and sorting of datasets, but it holds significant potential for future enhancements and scalability. One major improvement could be the addition of **advanced filtering options**, such as numeric range filters, date filters, and multi-condition filtering using checkboxes or sliders. The app could also incorporate **data visualization tools** like histograms, scatter plots, or bar charts

using packages such as ggplot2 or plotly to provide visual insights alongside the tabular data. Another valuable extension would be **summary statistics and analytical tools**, including mean, median, standard deviation, or custom calculations for selected columns. **User authentication and session management** can be added for personalized experiences and data privacy. Moreover, integration with **cloud storage or database systems** (e.g., Google Drive, MySQL) would allow for larger and more persistent datasets. Finally, making the app mobile-responsive and deployable on platforms like **shinyapps.io** or internal servers would broaden its accessibility and real-world applicability. These improvements would elevate the application from a simple data utility to a more comprehensive data exploration and decision-support tool.

CHAPTER 5

APPENDIX A – SOURCE CODE

```
library(shiny)
library(DT)
library(dplyr)

ui <- fluidPage(
  titlePanel("□ Simple Filtering and Sorting of Data in R"),
  sidebarLayout(
    sidebarPanel(
      fileInput("datafile", "Upload CSV File", accept = ".csv"),
      uiOutput("filter_ui"),
      uiOutput("sort_ui"),
      radioButtons("sort_dir", "Sort Direction", choices = c("Ascending" = "asc",
"Descending" = "desc")),
      downloadButton("download_data", "Download Filtered Data")
    ),
    mainPanel(
      DTOutput("table")
    )
  )
)

server <- function(input, output, session) {
  # Reactive dataset
  dataset <- reactive({
    req(input$datafile)
    read.csv(input$datafile$datapath, stringsAsFactors = FALSE)
  })
```

```
  # Dynamic filter UI
  output$filter_ui <- renderUI({
    req(dataset())
    selectInput("filter_col", "Filter by Column", choices = names(dataset()))
  })

  output$sort_ui <- renderUI({
    req(dataset())
    selectInput("sort_col", "Sort by Column", choices = names(dataset()))
  })

  # Filtered & sorted data
  filtered_data <- reactive({
    req(dataset(), input$filter_col, input$sort_col)
    data <- dataset()

    # Optional filter by value
    if (!is.null(input$filter_value) && input$filter_value != "") {
      data <- data[grepl(input$filter_value, data[[input$filter_col]], ignore.case =
TRUE), ]
    }

    # Sort
    if (input$sort_dir == "asc") {
      data <- data %>% arrange(.data[[input$sort_col]])
    } else {
      data <- data %>% arrange(desc(.data[[input$sort_col]]))
    }
```

```
    data
  })

  # Text input for filter value
  observeEvent(input$filter_col, {
    output$filter_value_ui <- renderUI({
      textInput("filter_value", paste("Filter value for", input$filter_col), "")
    })
  })
  output$filter_value_ui <- renderUI({}) # Initialize

  # Inject filter text input into UI
  insertUI(selector = "#filter_col", where = "afterEnd", ui =
uiOutput("filter_value_ui"))

  # Render the data table
  output$table <- renderDT({
    filtered_data()
  })

  # Download handler
  output$download_data <- downloadHandler(
    filename = function() {
      paste("filtered_data_", Sys.Date(), ".csv", sep = "")
    },
    content = function(file) {
      write.csv(filtered_data(), file, row.names = FALSE)
    }
  )
}
```

```
shinyApp(ui, server)
```

Appendix B – Screenshots

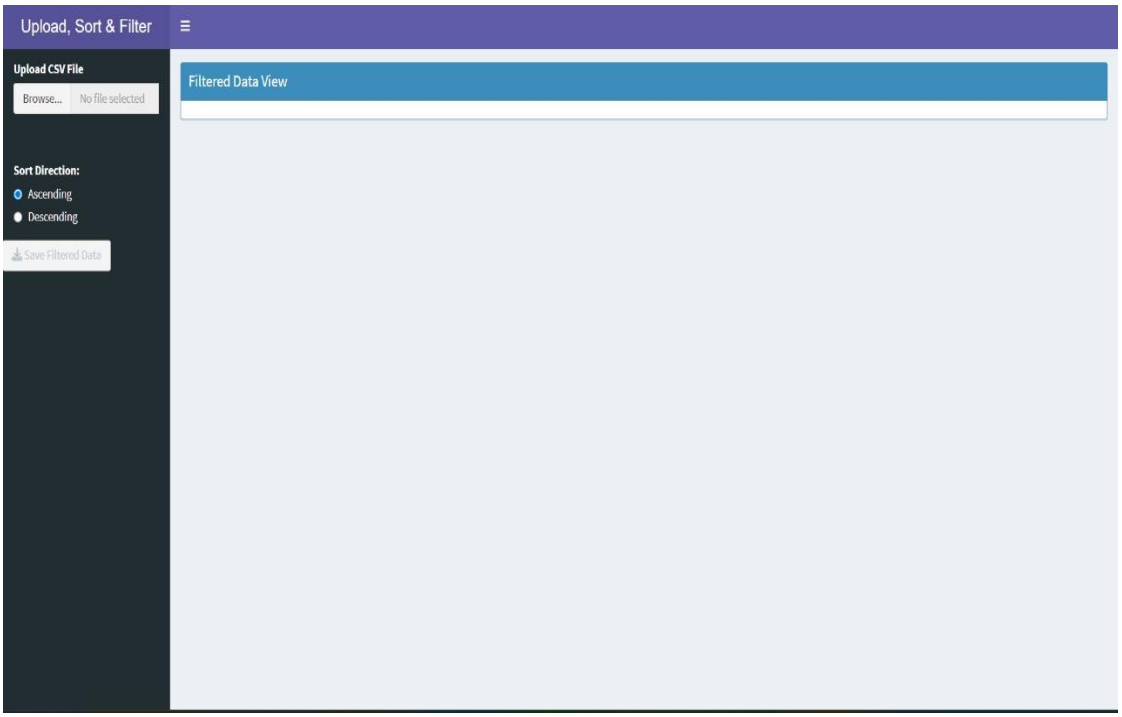


FIG 5.1 FILE UPLOAD

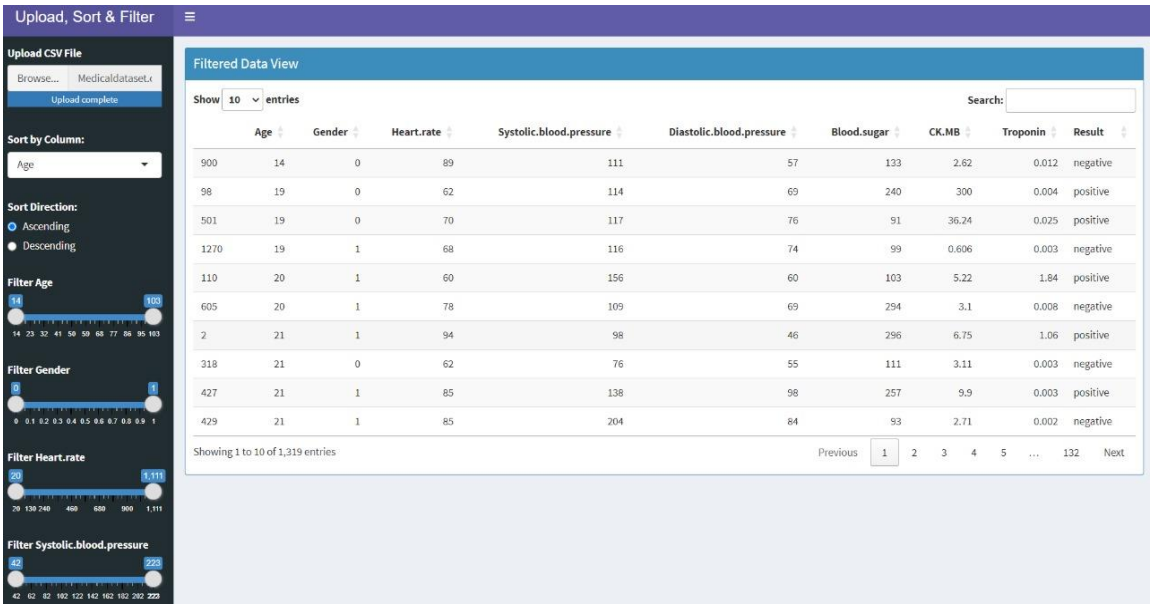


FIG 5.2 DATA VIEW

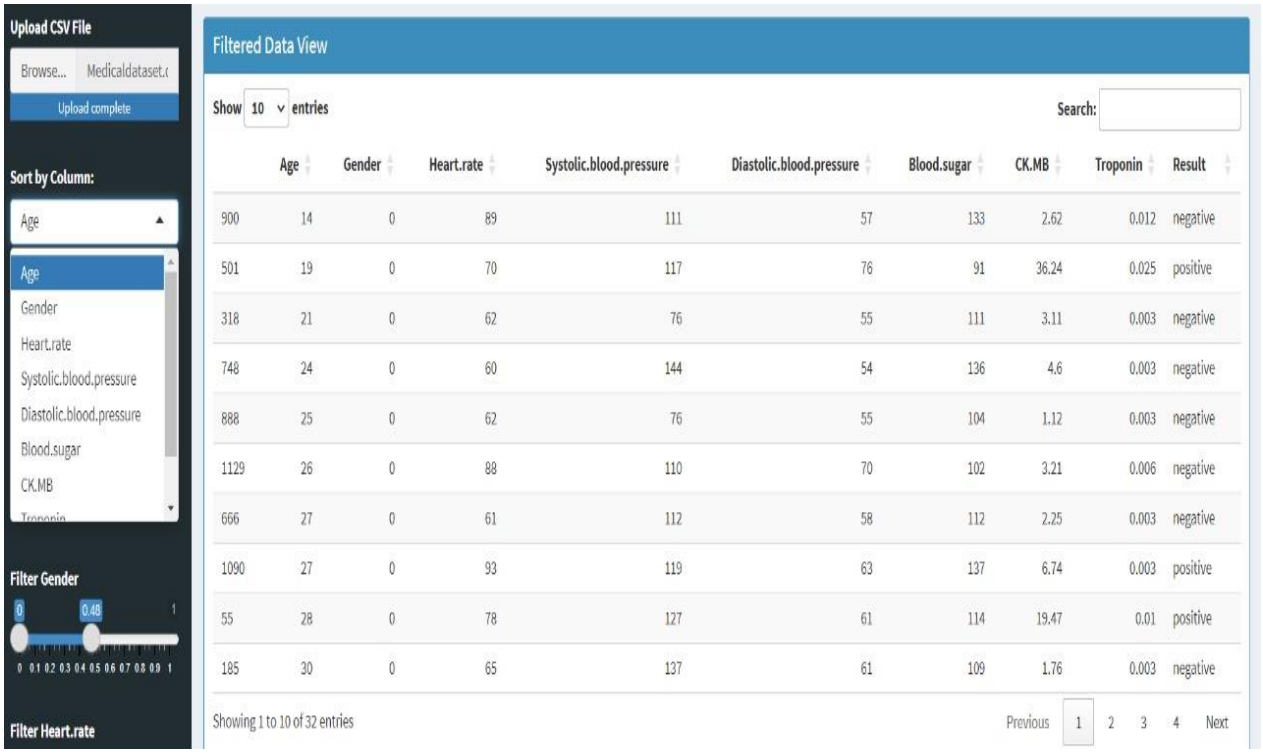


FIG 5.3COLUMN SELECTION

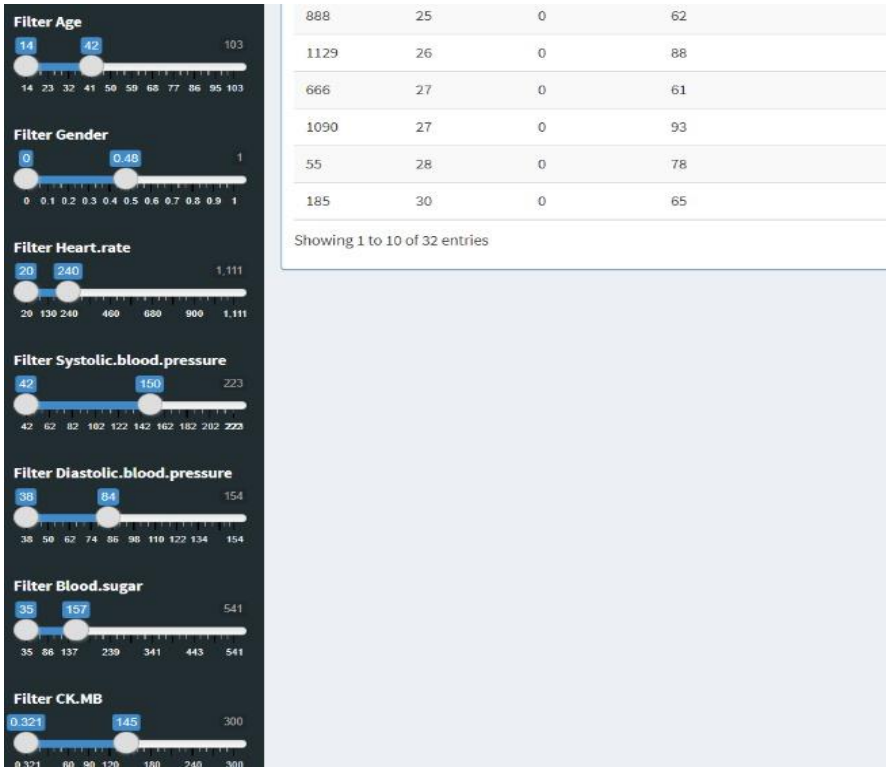


FIG 5.4 DORTING AND FILTERING

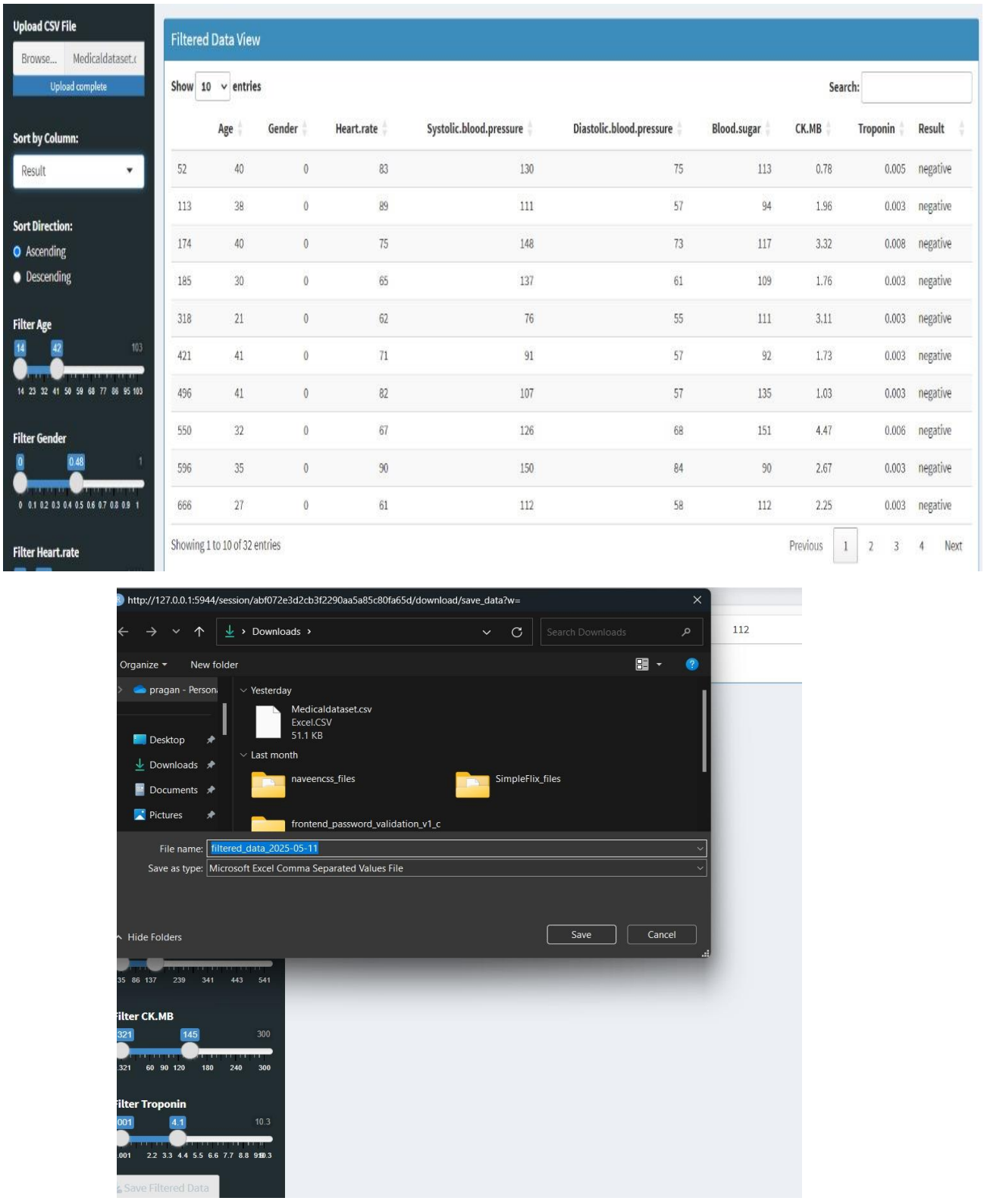


FIG 5.5 RESULT TABLE DISPLAY

REFERENCES

1. Wickham, H., François, R., Henry, L., & Müller, K. (2023). *dplyr: A Grammar of Data Manipulation*. R package version 1.1.4. https://cran.r-project.org/package=dplyr
2. Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., & Borges, B. (2023). *Shiny: Web Application Framework for R*. https://shiny.rstudio.com/
3. R Core Team. (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. https://www.r-project.org/
4. Xie, Y. (2022). *DT: A Wrapper of the JavaScript Library 'DataTables'*. R package version 0.28. https://rstudio.github.io/DT/
5. Grolemund, G., & Wickham, H. (2016). *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly Media. https://r4ds.hadley.nz/