****

**“UIC: UNIVERSITY INSTITUTE OF COMPUTING”**

**SUB: R PROGRAMMING LAB**

**Project:-**

**“Statistical Analysis and Visualization of Patient Health Metrics”**



**SUBMITTED BY:- SUBMITTED TO:-**

STUDENT NAME:- Sujal Thakur TEACHER NAME :- Shivani Chadha

BRANCH :- BCA(DATA SCIENCE) SUB CODE :- 24CAP-161

UID :- 24BCD10119 SECTION :- 24BCD-3(A)

SEMESTER :- SECOND

Statistical Analysis and Visualization of Patient Health Metrics Using R

# Abstract

This document outlines an R-based approach for analyzing and visualizing patient health metrics. The script is designed to process a small dataset of patients, computing key statistics, identifying extreme values, and generating informative plots. This workflow is useful for medical professionals, data analysts, and researchers looking to explore trends and outliers in patient data. The code is modular, making it easy to adapt to larger datasets or different health parameters.

# Introduction

The increasing availability of patient health data has created opportunities for deeper insights into individual and population-level health trends. This report presents a simple yet powerful implementation in R to demonstrate how basic patient metrics can be analyzed and visualized. Using a fabricated dataset, we explore metrics such as blood pressure, heart rate, and cholesterol levels across multiple patients. The goal is to extract meaningful statistical summaries and visualize them in a comprehensible format.

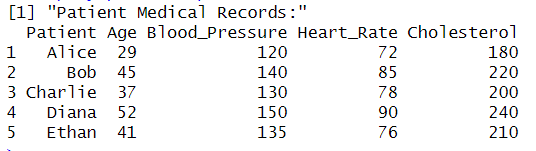
Methodology

The analysis begins with a dataset comprising patient names and corresponding health metrics. The script uses the tidyverse collection of packages, including dplyr, tidyr, and ggplot2. The main components of the methodology include:  
  
1. Data Initialization: A small dataset is created manually to simulate patient data.  
2. Display Function: A custom function prints the data table to verify its contents.  
3. Statistical Analysis: A function computes the mean, median, and standard deviation for each metric.  
4. Visualization: Bar plots, box plots, and histograms are generated to visualize distributions and comparisons.  
5. Identification of Extremes: The script identifies the patients with the highest and lowest values for each metric.  
  
Each function is modular, improving readability and reusability. The ggplot2 library provides a consistent framework for aesthetic and informative visualizations.

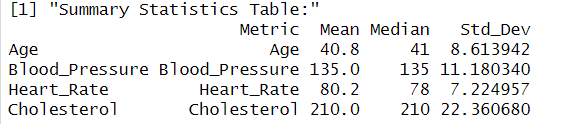
R Code

# Load necessary libraries  
library(ggplot2)  
library(tidyr)  
library(dplyr)  
  
# Sample dataset: Hospital patient metrics  
patient\_data <- data.frame(  
 Patient = c("Alice", "Bob", "Charlie", "Diana", "Ethan"),  
 Age = c(29, 45, 37, 52, 41),  
 Blood\_Pressure = c(120, 140, 130, 150, 135),  
 Heart\_Rate = c(72, 85, 78, 90, 76),  
 Cholesterol = c(180, 220, 200, 240, 210)  
)  
  
# Display patient data  
display\_patients <- function(data) {  
 print("Patient Medical Records:")  
 print(data)  
}

display\_patients(patient\_data)

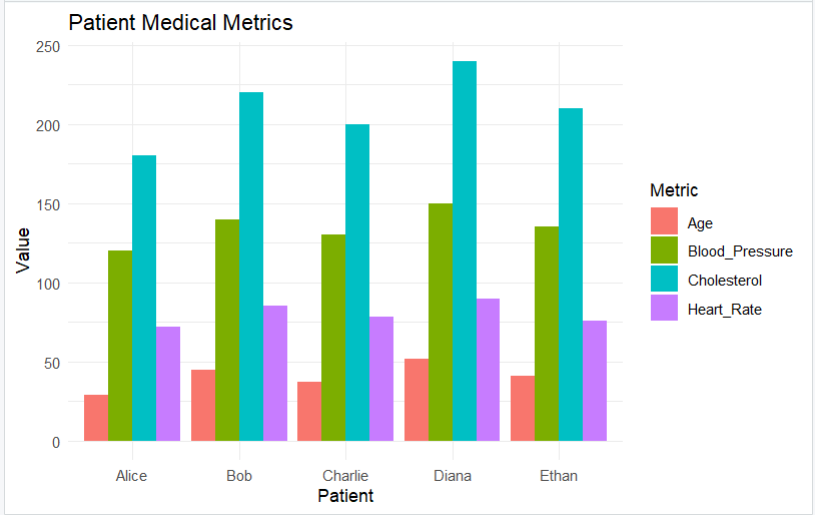


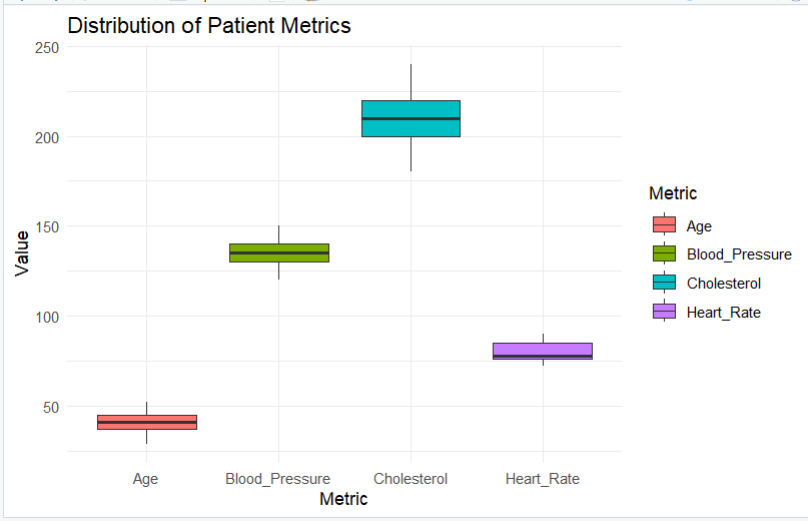
# Basic Statistical Analysis  
calculate\_statistics <- function(data) {  
 stats <- data.frame(  
 Metric = colnames(data)[-1],  
 Mean = sapply(data[-1], mean),  
 Median = sapply(data[-1], median),  
 Std\_Dev = sapply(data[-1], sd)  
 )  
 return(stats)  
}  
  
stats <- calculate\_statistics(patient\_data)  
print("Summary Statistics:")  
print(stats)  
  
# Summary Table  
summary\_table <- function(stats) {  
 print("Summary Statistics Table:")  
 print(stats)  
}  
  
summary\_table(stats)

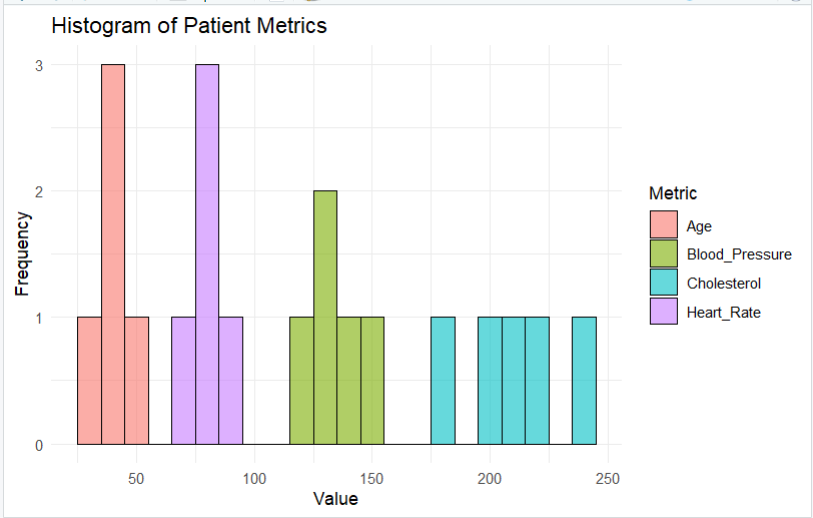


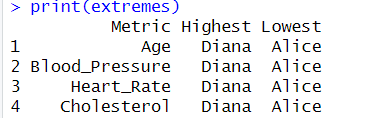
# Bar Plot for Patient Metrics  
bar\_plot <- function(data) {  
 melted\_data <- pivot\_longer(data, cols = -Patient, names\_to = "Metric", values\_to = "Value")  
 ggplot(melted\_data, aes(x = Patient, y = Value, fill = Metric)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 labs(title = "Patient Medical Metrics", x = "Patient", y = "Value", fill = "Metric") +  
 theme\_minimal()  
}

bar\_plot(patient\_data)



# Box Plot  
box\_plot <- function(data) {  
 melted\_data <- pivot\_longer(data, cols = -Patient, names\_to = "Metric", values\_to = "Value")  
 ggplot(melted\_data, aes(x = Metric, y = Value, fill = Metric)) +  
 geom\_boxplot() +  
 labs(title = "Distribution of Patient Metrics", x = "Metric", y = "Value") +  
 theme\_minimal()  
}  
  
box\_plot(patient\_data)

# Histogram  
histogram\_plot <- function(data) {  
 melted\_data <- pivot\_longer(data, cols = -Patient, names\_to = "Metric", values\_to = "Value")  
 ggplot(melted\_data, aes(x = Value, fill = Metric)) +  
 geom\_histogram(binwidth = 10, alpha = 0.6, position = "identity", color = "black") +  
 labs(title = "Histogram of Patient Metrics", x = "Value", y = "Frequency", fill = "Metric") +  
 theme\_minimal()  
}  
  
histogram\_plot(patient\_data)

# Identify Extremes  
identify\_extremes <- function(data) {  
 results <- lapply(names(data)[-1], function(metric) {  
 highest <- data[which.max(data[[metric]]), "Patient"]  
 lowest <- data[which.min(data[[metric]]), "Patient"]  
 return(data.frame(Metric = metric, Highest = highest, Lowest = lowest))  
 })  
 return(do.call(rbind, results))  
}  
  
extremes <- identify\_extremes(patient\_data)  
print("Highest and Lowest Values by Metric:")  
print(extremes)

Conclusion

This R-based approach offers a comprehensive, transparent, and reproducible methodology for analyzing and visualizing patient health metrics. By employing the tidyverse suite, the script ensures clean data manipulation and powerful visual storytelling. The ability to compute descriptive statistics such as mean, median, and standard deviation helps practitioners grasp central tendencies and variability across key health indicators. The visual tools—bar plots, box plots, and histograms—add significant value by making complex data patterns immediately interpretable. Additionally, identifying patients with the highest and lowest values in each metric supports personalized follow-up and clinical decision-making. Although the dataset is simplified for illustrative purposes, the structure is adaptable for real-world applications, enabling healthcare professionals to scale this model to more extensive and complex datasets. Ultimately, this document demonstrates how basic R programming skills can be leveraged to extract actionable insights from patient data, supporting both individual patient care and broader health analytics.