**Digital Health Summary**

(Diabetes Risk Factors)

Module GPH-03: Digital Health

Responsible: Prof. Dr. Dominik Böhler

*Deggendorf Institute of Technology*

*Faculty European Campus Rottal-Inn*

*Master of Global Public Health, 1st Semester*

Agyeman Serebouh Ordehyea Yaw

Pfarrkirchen, 31.01.2025

**USE CASE**

**Diabetes is a global public health burden**

The specific dataset used was on Diabetes. The data was used to compare risk factors like age, gender and bio markers such as cholesterol and LDL (low density lipoprotein) to diabetes.

Data was sourced from kaggle fairly and owned by (Kabir Olawale) uncleaned.

There were a total of 1099 rows and 14 columns. It was a large data sample collected from 800 patients and their bio markers.

Problems with duplications, missing values, incorrect naming and incorrect classifications were present . The data set didn’t specifiy which type of diabetes, however, diabetes type 2 is suggested based on the age group and lab markers.

**Specific Problems**

Duplication of ID numbers

Patient numbers not adding up

No classifications of values

Incorrect naming of columns using abbreviations

Capitalization problems

**SOLUTION & IMPLEMENTATION**

**Data Import and Initial Exploration**

I began by importing the raw dataset (diabetes\_unclean.csv) into R using the rio package. Since raw datasets often contain inconsistencies, missing values, and unstructured data, I performed an initial exploration using the skimr::skim() function to get an overview of the dataset, including missing values, distributions, and summary statistics.

**Data Cleaning and Preprocessing**

The next step involved renaming variables to more intuitive names using the rename() function within the dplyr package. Transformations included:

* Renaming columns (e.g., Chol to Cholesterol, AGE to Age\_years)
* Converting Age\_years into an integer format
* Creating categorical variables for different risk groups:
* Age Group: Segmented into age ranges (0-25, 26-35, etc.)
* BMI Group: Classified into Underweight, Normal, Overweight, Obesity, and Severe Obesity
* Cholesterol Group: Categorized into Normal, Borderline High, and High
* Triglyceride Group: Divided into Normal, Mildly Increased, Moderately Increased, and Very High
* LDL and HDL Groups: Segmented according to medical guidelines
* I removed duplicate rows and redundant patient numbers using distinct() while retaining unique patient IDs. The final cleaned dataset (diabetes\_clean) was structured with meaningful column names and groupings, making visualizations better.

**Data Summarization**

To classify diabetes distribution across different categories, I used R functions such as:

* nrow() and ncol() to check the number of rows and columns
* group\_by() and summarise() to count the number of patients in each category
* filter() and nrow() to count diabetic, pre-diabetic, and non-diabetic patients
* The summarized data allowed for a better understanding of how diabetes prevalence varied across age groups, BMI categories, cholesterol levels, and gender.

**Data Visualization**

To visualize the findings, I used ggplot2 to create various charts and plots:

* Stacked Bar Charts: Showed the proportion of diabetes status across BMI categories and gender
* Bar Plots: Displayed patient counts in different LDL, HDL, and cholesterol groups
* Scatter Plots: Represented the distribution of age among diabetic, pre-diabetic, and non-diabetic patients
* Using color schemes from RColorBrewer and viridis, I ensured accessibility and clarity in data representation.

**Procedure for Building the Shiny Diabetes Visualization App**

**Set Up the Environment**

* Install and load necessary R packages: shiny, ggplot2, and dplyr.
* Define the repository for package installation using options(repos = c(CRAN = "https://cloud.r-project.org/")).
* Load the Data
* Read the dataset using read.csv("diabetes\_clean.csv").
* Create the User Interface (UI)
* Use fluidPage() to structure the app layout.
* Add a titlePanel() to display the app title.
* Create a sidebar layout with sidebarLayout().
* Inside sidebarPanel(), add a selectInput() dropdown for users to choose different visualizations.
* Add conditionalPanel() elements with sliderInput() for filtering data when needed.
* Use mainPanel() to display the selected plot with plotOutput("selected\_plot").

**Define the Server Logic**

* Create a server function to process user input and generate plots.
* Use a reactive function (filtered\_data()) to filter the dataset based on user selections.
* Use renderPlot() to generate different plots based on the input$plot\_type value.
* Use ggplot2 functions to create visualizations such as bar charts and scatter plots.

**Run the Shiny App**

Use shinyApp(ui = ui, server = server) to launch the application.

**Conclusion**

Visualizations help establish relationships between bio markers and the diabetes status of the patient.

LINK: <https://ordehyea-diabetes-digitalhealth.shinyapps.io/ordehyea-diabetes-digital_health/>