**Practical Epidemiological Analysis Using ChatGPT and R**

*Jamalludin Ab Rahman, Muhammad ‘Adil Zainal Abidin*

Pre-conference Workshop, 12th National Public Health Conference. Everly Hotel Putrajaya. 7th July 2025

# 1. Introduction

Objective: Introduce the workshop objectives, tools, and dataset.

Key Points:

1. Role of ChatGPT in epidemiological analysis.
2. Overview of R as a statistical tool.
3. Dataset introduction: COVID-19 Linelist Deaths from the Ministry of Health Malaysia.
4. Using ChatGPT for debugging, generating R scripts, and exploring alternative approaches.

# 2. Data Setup and Exploration

Objective: Set up the environment and understand the dataset.

Activities:

1. Installing necessary R packages (tidyverse, lubridate, etc.).
2. Importing the dataset from the GitHub repository. [*https://raw.githubusercontent.com/MoH-*](https://raw.githubusercontent.com/MoH-)*Malaysia/covid19-public/refs/heads/main/epidemic/linelist/linelist\_deaths.csv*
3. Initial exploration: structure, summary statistics, and missing values.

# 3. Data Cleaning

Objective: Prepare the dataset for analysis.

Activities:

1. Check and handling missing data.
2. Creating derived variables, e.g., age groups (Children, Adults, Elderly).
3. Ensuring date formats are correct.
4. Format proper variable names, arrange variable values properly eg, Unvaccinated, First dose, Second Dose & Booster

# 4. Descriptive Analysis

Objective: Summarise key data features and visualise trends.

Activities:

1. Describe all variables in the dataset
2. Creating visualisations (e.g., bar plots).

# 5. Mortality Rates and Stratification

Objective: Calculate and analyse mortality rates.

Activities:

1. Mortality rate computation using population data.
2. Stratified analysis by age group, sex, BID status, co-morbidity and vaccination status

# 6. Trend Analysis

Objective: Explore mortality trends over time – epidemic curve

Activities:

1. Analysing daily mortality counts.
2. Visualising trends using line charts.

# 7. Advanced Techniques

Objective: Apply advanced epidemiological analysis methods.

Activities:

1. Treating BID as the main outcome - Hypothesis testing (e.g., Chi-square tests for categorical variables).
2. Logistic regression analysis to predict BID based on all explanatory variables

# 8. Ethical Use and reporting guideline of AI

Objective: Highlight the responsible use of AI in public health and transparency reporting based on current guideline.

Key Points:

1. Benefits and limitations of AI tools like ChatGPT.
2. Ensuring data privacy and avoiding bias in analyses.
3. The importance of human oversight in AI-generated outputs.
4. Transparency reporting of using AI.

# ===== 1. Introduction: Environment setup =====

setwd("~/Location-in-your-computer/Folder-for-all-the-analyses")

required\_packages <- c("tidyverse", "lubridate", "readr", "glue", "gtsummary", "arsenal", "gt")

for (pkg in required\_packages) {

if (!require(pkg, character.only = TRUE)) install.packages(pkg)

library(pkg, character.only = TRUE)

}

# ===== 2. Data Setup and Exploration =====

covid\_url <- "https://raw.githubusercontent.com/MoH-Malaysia/covid19-public/refs/heads/main/epidemic/linelist/linelist\_deaths.csv"

deaths <- read\_csv(covid\_url)

deaths %>%

tbl\_summary(missing = "always") %>%

modify\_caption("\*\*Initial Summary of COVID-19 Linelist Deaths Dataset\*\*") %>%

bold\_labels()

# ===== 3. Data Cleaning =====

deaths <- deaths %>%

mutate(

date = ymd(date),

date\_dose1 = ymd(date\_dose1),

date\_dose2 = ymd(date\_dose2),

date\_dose3 = ymd(date\_dose3),

age\_group = case\_when(

age < 18 ~ "Children",

age >= 18 & age < 60 ~ "Adult",

age >= 60 ~ "Elderly",

TRUE ~ NA\_character\_

),

vax\_status = case\_when(

!is.na(date\_dose3) ~ "Booster",

!is.na(date\_dose2) ~ "Fully Vaccinated",

!is.na(date\_dose1) ~ "Partially Vaccinated",

TRUE ~ "Unvaccinated"

),

gender = if\_else(male == 1, "Male", "Female"),

bid\_label = if\_else(bid == 1, "BID", "Non-BID"),

comorb\_label = if\_else(comorb == 1, "With Comorbidity", "No Comorbidity")

) %>%

mutate(

vax\_status = factor(vax\_status, levels = c("Unvaccinated", "Partially Vaccinated", "Fully Vaccinated", "Booster")),

age\_group = factor(age\_group, levels = c("Children", "Adult", "Elderly")),

gender = factor(gender, levels = c("Female", "Male"))

)

assign\_variable\_labels <- function(df) {

labels <- list(

age\_group = "Age Group",

vax\_status = "Vaccination Status",

gender = "Gender",

bid\_label = "BID Status",

comorb\_label = "Comorbidity",

state = "State"

)

for (var in names(labels)) {

attr(df[[var]], "label") <- labels[[var]]

}

df

}

deaths <- assign\_variable\_labels(deaths)

# ===== 4. Descriptive Analysis =====

deaths %>%

select(age\_group, gender, vax\_status, bid\_label, comorb\_label, state) %>%

tbl\_summary(missing = "always") %>%

modify\_caption("\*\*Summary of Cleaned COVID-19 Linelist Deaths Dataset\*\*") %>%

bold\_labels()

deaths %>% count(age\_group) %>%

ggplot(aes(age\_group, n, fill = age\_group)) +

geom\_col() +

labs(title = "COVID-19 Deaths by Age Group")

deaths %>% count(gender) %>%

ggplot(aes(gender, n, fill = gender)) +

geom\_col() +

labs(title = "COVID-19 Deaths by Gender")

deaths %>% count(vax\_status) %>%

ggplot(aes(vax\_status, n, fill = vax\_status)) +

geom\_col() +

labs(title = "COVID-19 Deaths by Vaccination Status")

deaths %>% count(bid\_label) %>%

ggplot(aes(bid\_label, n, fill = bid\_label)) +

geom\_col() +

labs(title = "COVID-19 Deaths by BID Status")

deaths %>% count(comorb\_label) %>%

ggplot(aes(comorb\_label, n, fill = comorb\_label)) +

geom\_col() +

labs(title = "COVID-19 Deaths by Comorbidity Status")

#====Using ChatGPT and Github copilot====

# Chatgpt in rstudio is recommended for coding explanation, debug and general knowledge

# Github copilot is recommended for real time code suggestion with ghost text, autocomplete style with proper sequence

# With github education eligible for github copilot free subscription, you can use copilot in Rstudio

#====Using Github copilot ====

#Copilot offers autocomplete-style suggestions as you code as “ghost text”

#This ghost-text is similar to the existing autocomplete available in RStudio but importantly is a generated suggestion rather than an exact auto-completion.

#1.Navigate to Tools > Global Options > Copilot.

#2.Check the box to “Enable GitHub Copilot”.

#3.Download and install the Copilot Agent components.

#4.Click the “Sign In” button.

#5.In the “GitHub Copilot: Sign in” dialog, copy the Verification Code.

#6.Navigate to or click on the link to https://github.com/login/device, paste the Verification Code and click “Continue”.

#7.GitHub will request the necessary permissions for GitHub Copilot. To approve these permissions, click “Authorize GitHub Copilot Plugin”.

#8.After the permissions have been approved, your RStudio IDE will indicate the currently signed in user.

#9.Close the Global Options dialogue, open a source file (.R, .py, .qmd, etc) and begin coding with Copilot!

#10.To disable GitHub Copilot Navigate to Tools > Global Options > Copilot and uncheck “Enable GitHub Copilot”, or

#11 (press tab to use the full suggestion)

#Example 1 bold\_labels

covid\_url <- "https://raw.githubusercontent.com/MoH-Malaysia/covid19-public/refs/heads/main/epidemic/linelist/linelist\_deaths.csv"

deaths <- read\_csv(covid\_url)

# Initial exploration (summary + missing)

deaths %>%

tbl\_summary(missing = "always") %>%

modify\_caption("\*\*Initial Summary of COVID-19 Linelist Deaths Dataset\*\*") %>%

#Example 2 - Plotting mortality rate by vaccination status

#Example 3- Compute max y for placing labels

#====Using chattr for link chatgpt in rstudio====

install.packages("chattr")

library(chattr)

library(shiny)

Sys.setenv("OPENAI\_API\_KEY" = "put your api here")

#get you free api key from open ai,Chattr will not properly installed or authenticated if you do not authenticate your API keys

#get your api from https://platform.openai.com/account/api-keys

chattr\_use("gpt4o") #select model 4o

chattr\_app(as\_job = FALSE)

#click addins, select open chat

#click the stop button of the application everytime finish chat and to resume running code

#close terminal once finish

#example how to generate R scripts with proper prompt

#====Generate an R script code =====

# Example chatgpt prompt is : Import dataset from https://raw.githubusercontent.com/MoH-Malaysia/covid19-public/refs/heads/main/epidemic/linelist/linelist\_deaths.csv into deaths dataset

deaths <- read\_csv("https://raw.githubusercontent.com/MoH-Malaysia/covid19-public/main/epidemic/linelist/linelist\_deaths.csv")

# Example copilot prompt is : summarize deaths dataset in table summary using dplyr and gtsummary

deaths\_summary <- deaths %>%

tbl\_summary(missing = "always") %>%

modify\_caption("\*\*Initial Summary of COVID-19 Linelist Deaths Dataset\*\*") %>%

bold\_labels()

#print the result

print(deaths\_summary)

# Example prompt is : Generate an R script code that run chi square using arsenal package using deaths dataset between bid variable and vax\_status variable

chi\_square\_result <- tableby(deaths$bid ~ deaths$vax\_status,

data = deaths,

stats = c("countpct", "chisq"))

summary(chi\_square\_result)

#====Debugging Rscript syntax issue====

#Please help debug this R script. The error I'm getting is: 'Error: unexpected symbol in: ")'. Here's the code:"

#fit\_bid < glm(bid ~ vax\_status + age\_group + gender + comorb + state, data = deaths, family = "binomial

#====Exploring alternative approaches====

#"My current script uses piping (%>%). Can you rewrite the same logic without using the pipe operator?". Here's the code: "fit\_bid %>% tbl\_regression(exp = TRUE) %>% modify\_caption("Logistic Regression Predicting BID Status") %>% modify\_header( label = "Variable", estimate = "OR", conf.low = "LL 95%CI", conf.high = "UL 95%CI", p.value = "p-value" ) %>% bold\_labels() "

tbl\_result <- tbl\_regression(fit\_bid, exp = TRUE)

tbl\_result <- modify\_caption(tbl\_result, "\*\*Logistic Regression Predicting BID Status\*\*")

tbl\_result <- modify\_header(

tbl\_result,

label = "\*\*Variable\*\*",

estimate = "\*\*OR\*\*",

conf.low = "\*\*LL 95%CI\*\*",

conf.high = "\*\*UL 95%CI\*\*",

p.value = "\*\*p-value\*\*"

)

tbl\_result <- bold\_labels(tbl\_result)

tbl\_result

# ===== 5. Mortality Rates and Stratification =====

# Load population data

pop\_url <- "https://storage.dosm.gov.my/population/population\_malaysia.csv"

pop\_raw <- read\_csv(pop\_url)

# Extract year from death dates

deaths <- deaths %>%

mutate(year = lubridate::year(date))

# Match population data to years present in death data

death\_years <- deaths %>%

distinct(year) %>%

pull(year)

pop\_summary <- pop\_raw %>%

filter(lubridate::year(date) %in% death\_years, sex == "both", ethnicity == "overall") %>%

mutate(age\_group = case\_when(

age %in% c("0-4", "5-9", "10-14") ~ "Children",

age == "15-19" ~ "Mixed",

age %in% c("20-24", "25-29", "30-34", "35-39", "40-44",

"45-49", "50-54", "55-59") ~ "Adult",

age %in% c("60-64", "65-69", "70-74", "75-79", "80-84", "85+") ~ "Elderly",

TRUE ~ NA\_character\_

))

# Summarise population by age group (across relevant years combined, or refine by year if needed)

pop\_base <- pop\_summary %>%

filter(age\_group != "Mixed") %>%

group\_by(age\_group) %>%

summarise(population = sum(population, na.rm = TRUE), .groups = "drop")

# Handle 15-19 split

pop\_15\_19 <- pop\_summary %>%

filter(age\_group == "Mixed") %>%

summarise(population = sum(population, na.rm = TRUE)) %>%

pull(population)

if (length(pop\_15\_19) == 1) {

pop\_base <- pop\_base %>%

mutate(population = case\_when(

age\_group == "Children" ~ population + (pop\_15\_19 \* 3 / 5 \* 1000),

age\_group == "Adult" ~ population + (pop\_15\_19 \* 2 / 5 \* 1000),

TRUE ~ population \* 1000

))

} else {

pop\_base <- pop\_base %>%

mutate(population = population \* 1000)

}

print(pop\_base)

# ===== Mortality rate by age group =====

mortality\_age <- deaths %>%

count(age\_group) %>%

left\_join(pop\_base, by = "age\_group") %>%

mutate(rate\_per\_100k = n / population \* 100000)

print(mortality\_age)

mortality\_age %>%

mutate(age\_group = factor(age\_group, levels = c("Children", "Adult", "Elderly"))) %>%

ggplot(aes(age\_group, rate\_per\_100k, fill = age\_group)) +

geom\_col() +

labs(title = "Mortality Rate per 100,000 by Age Group", x = "Age Group", y = "Rate per 100,000") +

theme\_minimal()

# ===== Mortality rate by gender =====

# Replace with actual population data by gender if available

pop\_sex <- tibble(gender = c("Female", "Male"), population = c(16000000, 16000000)) # Example only

mortality\_gender <- deaths %>%

count(gender) %>%

left\_join(pop\_sex, by = "gender") %>%

mutate(rate\_per\_100k = n / population \* 100000)

print(mortality\_gender)

mortality\_gender %>%

ggplot(aes(gender, rate\_per\_100k, fill = gender)) +

geom\_col() +

labs(title = "Mortality Rate per 100,000 by Gender", x = "Gender", y = "Rate per 100,000") +

theme\_minimal()

# ===== Mortality rate by BID status =====

pop\_total <- sum(pop\_base$population)

mortality\_bid <- deaths %>%

count(bid\_label) %>%

mutate(rate\_per\_100k = n / pop\_total \* 100000)

print(mortality\_bid)

mortality\_bid %>%

ggplot(aes(bid\_label, rate\_per\_100k, fill = bid\_label)) +

geom\_col() +

labs(title = "Mortality Rate per 100,000 by BID Status", x = "BID Status", y = "Rate per 100,000") +

theme\_minimal()

# ===== Mortality rate by comorbidity =====

mortality\_comorb <- deaths %>%

count(comorb\_label) %>%

mutate(rate\_per\_100k = n / pop\_total \* 100000)

print(mortality\_comorb)

mortality\_comorb %>%

ggplot(aes(comorb\_label, rate\_per\_100k, fill = comorb\_label)) +

geom\_col() +

labs(title = "Mortality Rate per 100,000 by Comorbidity", x = "Comorbidity Status", y = "Rate per 100,000") +

theme\_minimal()

# ===== Mortality rate by vaccination status =====

mortality\_vax <- deaths %>%

count(vax\_status) %>%

mutate(rate\_per\_100k = n / pop\_total \* 100000)

print(mortality\_vax)

mortality\_vax %>%

ggplot(aes(vax\_status, rate\_per\_100k, fill = vax\_status)) +

geom\_col() +

labs(title = "Mortality Rate per 100,000 by Vaccination Status", x = "Vaccination Status", y = "Rate per 100,000") +

theme\_minimal()

# ===== 6. Trend Analysis =====

max\_y <- deaths %>%

count(date) %>%

summarise(max\_n = max(n, na.rm = TRUE)) %>%

pull(max\_n)

wave\_dates <- tibble(

wave = c("Wave 1", "Wave 2", "Wave 3", "Wave 4"),

date = as.Date(c("2020-01-25", "2020-02-27", "2020-09-01", "2021-06-01"))

)

deaths %>%

count(date) %>%

ggplot(aes(date, n)) +

geom\_line(color = "red") +

geom\_vline(data = wave\_dates, aes(xintercept = as.numeric(date)), linetype = "dashed", color = "blue") +

geom\_text(data = wave\_dates, aes(x = date, y = max\_y, label = wave), angle = 90, vjust = -0.5, hjust = 1, color = "blue") +

labs(title = "Daily COVID-19 Deaths Over Time with Wave Markers", x = "Date", y = "Daily Deaths") +

theme\_minimal()

# ===== 7. Advanced Techniques =====

# --- Descriptive relationship between BID and key variables among the dead ---

table\_biv <- tableby(

bid ~ vax\_status + age\_group + gender + comorb\_label + state,

data = deaths

)

summary(table\_biv, text = TRUE)

write2word(table\_biv, file = "Table - BID status.docx")

# --- Assign wave period to each death ---

deaths <- deaths %>%

mutate(wave = case\_when(

date >= as.Date("2020-01-25") & date <= as.Date("2020-02-26") ~ "Wave 1",

date >= as.Date("2020-02-27") & date <= as.Date("2020-07-31") ~ "Wave 2",

date >= as.Date("2020-09-01") & date <= as.Date("2021-03-31") ~ "Wave 3",

date >= as.Date("2021-06-01") & date <= as.Date("2022-01-31") ~ "Wave 4",

TRUE ~ "Other"

))

# --- Logistic regression to model BID status among the dead ---

fit\_bid <- glm(

bid ~ vax\_status + age\_group + gender + comorb + state + wave,

data = deaths,

family = "binomial"

)

fit\_bid %>%

tbl\_regression(exp = TRUE) %>%

modify\_caption("\*\*Logistic Regression Predicting BID Status among Deaths\*\*") %>%

modify\_header(

label = "\*\*Variable\*\*",

estimate = "\*\*OR\*\*",

conf.low = "\*\*LL 95%CI\*\*",

conf.high = "\*\*UL 95%CI\*\*",

p.value = "\*\*p-value\*\*"

) %>%

bold\_labels()

# --- BID trends over time by vaccination status ---

deaths %>%

mutate(period = format(date, "%Y-%m")) %>%

count(period, vax\_status, bid\_label) %>%

ggplot(aes(period, n, fill = bid\_label)) +

geom\_col(position = "fill") +

facet\_wrap(~ vax\_status, ncol = 1) +

scale\_y\_continuous(labels = scales::percent) +

labs(

title = "Proportion of COVID-19 Deaths that were BID by Vaccination Status Over Time",

x = "Month",

y = "Proportion"

) +

theme\_minimal() +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

#==== 8.Key Ethical points====

#Objective: Highlight the responsible use of AI in public health and transparency reporting based on current guideline.

#Key Points:

#1. Benefits and limitations of AI tools like ChatGPT.

#2. Ensuring data privacy and avoiding bias in analyses.

#3. The importance of human oversight in AI-generated outputs.

#4. Transparency reporting of using AI.

#1. Benefits and Limitations of AI Tools Like ChatGPT

#AI tools can enhance productivity, improve data analysis, and support health communication. ChatGPT, for instance, can assist in summarizing research, drafting reports, and generating health education content.

#However, limitations include potential inaccuracies, hallucinated outputs, and lack of contextual awareness. AI models may also reflect biases present in their training data.

#2. Ensuring Data Privacy and Avoiding Bias

#Ethical AI use must comply with data protection regulations (e.g., HIPAA, GDPR). Sensitive or identifiable health data should never be input into generative AI tools.

#Bias in AI outputs must be mitigated by critically assessing data sources, ensuring diverse training inputs, and involving domain experts in evaluation processes.

#3. Importance of Human Oversight

#AI-generated content should be reviewed and validated by qualified professionals. Final decisions—especially in clinical or public health contexts—must remain under human authority to maintain accountability and context-appropriate judgment.

#4. Transparency in AI Use Reporting

#When AI tools contribute to research or public health documentation, their use must be clearly disclosed. This includes:

#Stating the name and version of the tool (e.g., OpenAI ChatGPT, GPT-4).

#Describing how AI was used (e.g., content generation, data analysis assistance).

#Clarifying that outputs were reviewed and edited by humans.

#Example statement for research reporting:

# “Portions of this manuscript were generated with the assistance of ChatGPT (OpenAI), and subsequently reviewed and edited by the authors for accuracy and context.”

#Local References

#1. https://mmc.gov.my/wp-content/uploads/2025/04/Ethical-Use-of-Artificial-Intelligence-in-Medical-Practice.pdf