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-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")

# ===== 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))
```