**Appendix 1**

**R code for generating test\_score\_data**

# Set seed for reproducibility

set.seed(123)

# Number of students

n <- 1000

# Generate age, sex, hours studied, and tutor status

age <- round(runif(n, min = 20, max = 30)) # Uniform distribution for age (20 to 30)

sex <- sample(c("Male", "Female"), n, replace = TRUE, prob = c(0.5, 0.5)) # Randomly sample Male and Female

hours\_studied <- round(runif(n, min = 0, max = 100)) # Random uniform distribution for hours studied (0 to 100)

tutor <- sample(c("Yes", "No"), n, replace = TRUE, prob = c(0.3, 0.7)) # Randomly assign tutor status (30% Yes, 70% No)

# Function to generate test scores based on inputs

generate\_test\_score <- function(age, sex, hours\_studied, tutor) {

base\_score <- 30 # Base score (minimum score)

age\_effect <- (age - 20) \* 2 # Older students get slightly higher score (up to 20 points more)

sex\_effect <- ifelse(sex == "Female", 10, 0) # Females get +10 score advantage

study\_effect <- 0.015 \* hours\_studied^2 - 0.0001 \* hours\_studied^3 # Adjusted quadratic relationship

tutor\_effect <- ifelse(tutor == "Yes", 15, 0) # Students with tutor get +15 score advantage

# Calculate final test score

test\_score <- base\_score + age\_effect + sex\_effect + study\_effect + tutor\_effect

# Ensure test score is at least 30 and at most 100

test\_score <- pmax(test\_score, 30)

test\_score <- pmin(test\_score, 100)

return(test\_score)

}

# Generate test score using the function

test\_score <- generate\_test\_score(age, sex, hours\_studied, tutor)

# Create data frame

test\_score\_data <- data.frame(age = age, sex = sex, hours\_studied = hours\_studied, tutor = tutor, test\_score = test\_score)

# Print first few rows of the data frame

head(test\_score\_data)

**Appendix 2**

**R code for generating salary\_data**

# Install necessary packages

#install.packages("dplyr")

# Load necessary libraries

library(dplyr)

# Set seed for reproducibility

set.seed(123)

# Number of individuals

n <- 2500

# Generate age, sex, university degree status, and field of work

age <- round(runif(n, min = 20, max = 60)) # Uniform distribution for age (20 to 60)

sex <- sample(c("Male", "Female"), n, replace = TRUE, prob = c(0.5, 0.5)) # Randomly sample Male and Female

university\_degree <- sample(c("Yes", "No"), n, replace = TRUE, prob = c(0.7, 0.3)) # 70% Yes, 30% No

field\_of\_work <- sample(c("Healthcare", "Engineering", "Law", "Business", "Arts", "Finance"), n, replace = TRUE)

# Function to generate annual salary based on inputs

generate\_annual\_salary <- function(age, sex, university\_degree, field\_of\_work) {

base\_salary <- 20 # Base salary in thousands

# Age effect: Piecewise approximation capturing rapid increase, slower growth, and eventual plateau or small decrease

age\_effect <- ifelse(age <= 25, 8 \* (age - 20),

ifelse(age <= 40, 50 + 2 \* (age - 25),

90 - 1 \* (age - 40)))

# Sex effect: Adjust for gender pay gap (not real-world accurate, just for example)

sex\_effect <- ifelse(sex == "Male", 10, 0)

# University degree effect: Individuals with a degree earn more

degree\_effect <- ifelse(university\_degree == "Yes", 15, 0)

# Field of work effect: Different fields have different average salaries

work\_effect <- case\_when(

field\_of\_work == "Healthcare" ~ min(80, 30 + rnorm(1, sd = 15)),

field\_of\_work == "Engineering" ~ min(800, 60 + rnorm(1, sd = 30)),

field\_of\_work == "Law" ~ min(800, 50 + rnorm(1, sd = 20)),

field\_of\_work == "Business" ~ min(80, 40 + rnorm(1, sd = 10)),

field\_of\_work == "Arts" ~ min(80, 25 + rnorm(1, sd = 10)),

field\_of\_work == "Finance" ~ min(800, 100 + rnorm(1, sd = 40))

)

# Introduce moderate variability

variability <- rnorm(n, mean = 0, sd = 12)

# Calculate final annual salary

annual\_salary <- base\_salary + age\_effect + sex\_effect + degree\_effect + work\_effect + variability

return(annual\_salary)

}

# Generate annual salary using the function

annual\_salary <- generate\_annual\_salary(age, sex, university\_degree, field\_of\_work)

# Create data frame

salary\_data <- data.frame(age = age, sex = sex, university\_degree = university\_degree, field\_of\_work = field\_of\_work, annual\_salary = annual\_salary)

# Print first few rows of the data frame

head(salary\_data)

**Appendix 3**

**R code for generating social\_media\_usage\_data**

# Load necessary libraries

library(dplyr)

# Set seed for reproducibility

set.seed(123)

# Generate synthetic data for 2000 kids

n\_people <- 2000

ages <- sample(seq(11, 18, by = 0.5), n\_people, replace = TRUE)

days <- sample(c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"), n\_people, replace = TRUE)

# Create a data frame with social media usage

social\_media\_usage\_data <- data.frame(age = ages, day = days)

social\_media\_usage\_data$usage <- with(social\_media\_usage\_data,

ifelse(age >= 13.5 & age <= 16.5, ifelse(day %in% c("Friday", "Saturday", "Sunday"), rnorm(n\_people, mean = 450, sd = 50), rnorm(n\_people, mean = 250, sd = 50)), rnorm(n\_people, mean = 100, sd = 30)))

# Ensure no negative usage values

social\_media\_usage\_data$usage <- pmax(social\_media\_usage\_data$usage, 0)

# Print first few rows of the data frame

head(social\_media\_usage\_data)

**R code for generating patient\_data**

# Load necessary libraries

library(dplyr)

set.seed(123) # For reproducibility

n <- 25000 # number of patients

# Simulate 10 exposure variables

patient\_data <- data.frame(

age = round(rnorm(n, mean = 60, sd = 12)), # continuous

sex = sample(c("Male", "Female"), n, replace = TRUE), # categorical

smoker = rbinom(n, 1, 0.3), # binary

bmi = round(rnorm(n, mean = 27, sd = 5), 1), # continuous

diabetes = rbinom(n, 1, 0.15), # binary

comorbidity\_count = rpois(n, lambda = 2), # count

surgery\_type = sample(c("Minor", "Major", "Emergency"), n,

replace = TRUE,

prob = c(0.5, 0.4, 0.1)), # categorical

days\_in\_hospital\_last\_year = rpois(n, lambda = 5), # count

previous\_surgeries = rbinom(n, 3, 0.4), # count

insurance\_type = sample(c("Public", "Private", "None"), n,

replace = TRUE,

prob = c(0.5, 0.45, 0.05)) # categorical

)

# Create dummy variables for modeling

patient\_data <- patient\_data %>%

mutate(

sex\_female = ifelse(sex == "Female", 1, 0),

surg\_emerg = ifelse(surgery\_type == "Emergency", 1, 0),

surg\_major = ifelse(surgery\_type == "Major", 1, 0),

ins\_none = ifelse(insurance\_type == "None", 1, 0),

ins\_public = ifelse(insurance\_type == "Public", 1, 0)

)

# Adjusted linear predictor: lower the intercept to bring down predicted probabilities

logit <- with(patient\_data,

-6.3 +

0.03 \* age + # weak

0.5 \* sex\_female + # moderate

1.0 \* smoker + # strong

0.04 \* bmi + # weak/moderate

1.2 \* diabetes + # strong

0.25 \* comorbidity\_count + # moderate

1.5 \* surg\_emerg + # strong

0.7 \* surg\_major + # moderate

0.1 \* days\_in\_hospital\_last\_year + # weak/moderate

0.7 \* ins\_none + # strong

0.3 \* ins\_public + # moderate

0.3 \* previous\_surgeries # moderate

)

# Convert logit to probability

prob <- 1 / (1 + exp(-logit))

# Simulate binary outcome

patient\_data$hospitalized\_30d <- rbinom(n, 1, prob)

# Remove modeling helper variables

patient\_data <- patient\_data %>%

select(-sex\_female, -surg\_emerg, -surg\_major, -ins\_none, -ins\_public)

# Print first few rows of the data frame

head(patient\_data)

**Appendix 4**

**R code for generating Figure 4.1.1 (Scatter plot)**

# Install necessary packages

#install.packages("ggplot2")

# Load necessary libraries

library(ggplot2)

# Create a scatter plot of test score vs. hours studied

ggplot(test\_score\_data, aes(x = hours\_studied, y = test\_score)) +

geom\_point(alpha = 0.5) + # Scatter plot with transparency

labs(title = "Test Score vs. Hours Studied", x = "Hours Studied", y = "Test Score") +

theme\_minimal() # Minimal theme for clean appearance

**R code for generating Figure 4.1.2 (Scatter plot)**

# Load necessary libraries

library(ggplot2)

# Create a scatter plot of test score vs. hours studied, by tutor status

ggplot(test\_score\_data, aes(x = hours\_studied, y = test\_score, color = tutor)) +

geom\_point(alpha = 0.5) + # Scatter plot with transparency

labs(title = "Test Score vs. Hours Studied, by Tutor Status", x = "Hours Studied", y = "Test Score", color = "Tutor Status") +

theme\_minimal() # Minimal theme for clean appearance

**R code for generating Figure 4.1.3 (Facet plot)**

# Load necessary libraries

library(ggplot2)

# Create a customized facet plot with black axis labels

p <- ggplot(test\_score\_data, aes(x = hours\_studied, y = test\_score, color = tutor)) +

geom\_point(alpha = 0.6, size = 2) +

facet\_grid(sex ~ tutor) +

labs(

title = "Test Scores by Hours Studied, Sex, and Tutor Status",

subtitle = "Comparison of test scores across different groups",

x = "Hours Studied",

y = "Test Score",

color = "Tutor Status"

) + theme\_minimal() +

theme(

plot.title = element\_text(hjust = 0.5, size = 16, face = "bold"),

plot.subtitle = element\_text(hjust = 0.5, size = 12, face = "italic"),

axis.title.x = element\_text(size = 12, color = "black"),

axis.title.y = element\_text(size = 12, color = "black"),

strip.text = element\_text(size = 12, face = "bold"),

panel.grid.major = element\_line(color = "grey80"),

panel.grid.minor = element\_blank(),

legend.position = "top"

)

# Print the plot

print(p)

**R code for generating Figure 4.1.4 (Jittered scatter plot)**

# Load necessary libraries

library(ggplot2)

ggplot(patient\_data, aes(x = bmi, y = hospitalized\_30d, color = factor(smoker))) +

geom\_jitter(height = 0.05, width = 0, alpha = 0.3) +

scale\_y\_continuous(breaks = c(0, 1)) +

scale\_color\_manual(values = c("0" = "steelblue", "1" = "red")) +

labs(

title = "Hospitalization Within 30 Days by BMI and Smoking Status",

x = "BMI",

y = "Hospitalized (0 = No, 1 = Yes)",

color = "Smoker"

) +

theme\_minimal()

**R code for generating Figure 4.2 (Line graph)**

# Load necessary libraries

library(ggplot2)

# Calculate average test score by age

avg\_test\_score\_by\_age <- aggregate(test\_score ~ age, data = test\_score\_data,

FUN = mean)

# Plotting with ggplot2

ggplot(avg\_test\_score\_by\_age, aes(x = age, y = test\_score)) +

geom\_line() + # Line graph

geom\_point() + # Add points to the line graph

labs(title = "Average Test Score by Age",

x = "Age",

y = "Average Test Score") +

scale\_y\_continuous(limits = c(30, 100)) + # Set y-axis limits from 30 to 100

theme\_minimal() # Minimal theme for clean appearance

**R code for generating Figure 4.3 (Bar plot)**

# Load necessary libraries

library(ggplot2)

library(dplyr)

# Calculate hospitalization rate by insurance type

ins\_plot <- patient\_data %>%

group\_by(insurance\_type) %>%

summarise(

hospitalization\_rate = mean(hospitalized\_30d),

count = n()

)

# Bar plot with value labels

ggplot(ins\_plot, aes(x = insurance\_type, y = hospitalization\_rate, fill = insurance\_type)) +

geom\_col() +

geom\_text(aes(label = scales::percent(hospitalization\_rate, accuracy = 0.1)),

vjust = -0.3, size = 4.5) +

scale\_y\_continuous(labels = scales::percent, limits = c(0, max(ins\_plot$hospitalization\_rate) + 0.05)) +

labs(

title = "Hospitalization Rates by Insurance Type",

x = "Insurance Type",

y = "Hospitalization Rate"

) +

theme\_minimal() +

theme(legend.position = "none")

**R code for generating Figure 4.4 (Pie chart)**

# Load necessary libraries

library(ggplot2)

library(dplyr)

library(scales)

# Ensure salary is in full dollars

salary\_data <- salary\_data %>%

mutate(salary\_dollars = annual\_salary \* 1000)

# Define bin settings

min\_bin <- 35000

bin\_size <- 45000

max\_salary <- max(salary\_data$salary\_dollars, na.rm = TRUE)

last\_edge <- ceiling((max\_salary - min\_bin) / bin\_size) \* bin\_size + min\_bin

# Create bin edges and labels

bin\_edges <- seq(min\_bin, last\_edge, by = bin\_size)

bin\_labels <- c(

paste0(

"$", format(bin\_edges[-length(bin\_edges)], big.mark = ","),

" – $", format(bin\_edges[-1] - 1, big.mark = ",")

),

paste0("$", format(tail(bin\_edges, 1), big.mark = ","), " +")

)

# Cut into salary brackets with fixed labels

salary\_data <- salary\_data %>%

mutate(

salary\_bracket\_45k = cut(

salary\_dollars,

breaks = c(bin\_edges, Inf),

labels = bin\_labels,

right = FALSE,

include.lowest = TRUE

)

)

# Summarize frequency and proportion

salary\_dist\_45k <- salary\_data %>%

filter(!is.na(salary\_bracket\_45k)) %>%

count(salary\_bracket\_45k) %>%

mutate(

proportion = n / sum(n),

label\_with\_pct = paste0(

as.character(salary\_bracket\_45k),

" (", percent(proportion, accuracy = 1), ")"

)

)

# Plot using the original bin as the fill and custom labels in the legend

ggplot(salary\_dist\_45k, aes(x = "", y = n, fill = salary\_bracket\_45k)) +

geom\_col(width = 1, color = "white") +

coord\_polar(theta = "y") +

scale\_fill\_manual(

values = scales::hue\_pal()(nrow(salary\_dist\_45k)),

labels = salary\_dist\_45k$label\_with\_pct,

breaks = salary\_dist\_45k$salary\_bracket\_45k

) +

labs(

title = "Salary Distribution (Binned by $45,000)",

fill = "Salary Range (Proportion)"

) +

theme\_void() +

theme(legend.position = "right")

**Appendix 5**

**Quadratic formula – Deeper dive**

If you've used the quadratic formula before:

,

you may have wondered where this formula comes from? Let’s break down it’s derivation step by step using algebra and a method called completing the square.

Start with the general quadratic equation:

(where )

The aim is to find the value(s) of that satisfies this equation.

Step 1. Make the coefficient of equal to 1 (by dividing both side by ):

Step 2. Move the constant to the right side, so only the variable terms ( terms) are on the left side:

Step 3. Complete the square by taking half of the coefficient of (half of is ), squaring it (), and adding it to both sides:

Step 4. Factor the left-hand side so it is now a perfect square:

Step 5. Take the square root of both sides (don’t forget the ± symbol!):

Step 6. Solve for :

This formula works to solve for under any quadratic equation of the form .

**R code for generating Figure 5.1.1**

# Load necessary libraries

library(ggplot2)

# Define the data for months and total savings

months <- 0:12

savings <- 30 + 10 \* months

# Create a data frame for plotting

savings\_data <- data.frame(Months = months, Savings = savings)

# Create the plot

ggplot(savings\_data, aes(x = Months, y = Savings)) +

geom\_line(color = "blue", linewidth = 1.2) +

geom\_point(color = "red", size = 2) +

geom\_hline(yintercept = 120, linetype = "dashed", color = "darkgreen") +

annotate("text", x = 1, y = 120, label = "Game Cost ($120)", vjust = -1,

hjust = 0.5, color = "darkgreen", size = 3.5) +

labs(

title = "Shaan's Savings Over Time",

x = "Months",

y = "Total Savings($)",

caption = "The dashed green line represents the cost of the video game. The plot shows how Shaan's savings grow over time."

) +

theme\_minimal() +

scale\_x\_continuous(breaks = seq(0, 12, by = 1)) +

scale\_y\_continuous(breaks = seq(0, 140, by = 10))

**R code for generating Figure 5.1.2**

# Load necessary libraries

library(ggplot2)

# Define time from 0 to 80 minutes

time <- 0:80 # 81 points including zero

initial\_sand <- 36 # Initial amount of sand in cups

# Calculate the amount of sand in the bucket over time

sand\_amount <- initial\_sand + (1 - 1.5) \* time

# Create a data frame for plotting

sand\_data <- data.frame(

Time = time,

Sand\_Amount = sand\_amount

)

# Create the plot

ggplot(sand\_data, aes(x = Time, y = Sand\_Amount)) +

geom\_line(color = "blue", linewidth = 1) + # Line for sand amount over time

geom\_point(size = 2) + # Points at each time step

labs(title = "Amount of Sand in the Bucket Over Time", x = "Time (minutes)", y = "Amount of Sand (cups)") +

theme\_minimal() +

geom\_hline(yintercept = 0, linetype = "dashed", color = "red") +

annotate("text", x = 70, y = 0.5, label = "Bucket is empty", hjust = 1.1, color = "red", size = 3.5) # Text annotation

**R code for generating Figure 5.1.3**

# Load necessary libraries

library(ggplot2)

# Define the initial variables

weeks <- 0:20

riya\_distance <- 3 + 0.8 \* weeks

shaan\_distance <- 8 + 0.4 \* weeks

distance\_data <- data.frame(weeks = weeks, riya\_distance = riya\_distance, shaan\_distance = shaan\_distance)

# Create the plot

ggplot(distance\_data) +

geom\_line(aes(x = weeks, y = riya\_distance, color = "Riya's Distance"), size = 1) +

geom\_line(aes(x = weeks, y = shaan\_distance, color = "Shaan's Distance"), size = 1) +

geom\_point(aes(x = weeks, y = riya\_distance, color = "Riya's Distance"), size = 2) +

geom\_point(aes(x = weeks, y = shaan\_distance, color = "Shaan's Distance"), size = 2) +

scale\_color\_manual(values = c("Riya's Distance" = "blue", "Shaan's Distance" = "green")) +

scale\_x\_continuous(breaks = seq(0, 20, 1)) +

labs(title = "Riya and Shaan's Running Progress Over Time", x = "Weeks", y = "Running Distance (km)", color = "Legend") +

theme\_minimal() +

theme(legend.position = "right", legend.justification = c(1, 0.5))

**R code for generating Figure 5.1.4**

# Load necessary libraries

library(ggplot2)

# Define the number of weeks

weeks <- 0:20 # Plot for 21 weeks

# Calculate the money for each person over the weeks

shaan\_money <- 130 - 5 \* weeks

riya\_money <- 25 + 10 \* weeks

disha\_money <- 50 + 10 \* weeks

# Create a data frame

money\_data <- data.frame(

weeks = weeks, Shaan = shaan\_money, Riya = riya\_money, Disha = disha\_money

)

# Reshape the data frame to long format

money\_data\_long <- data.frame(

weeks = rep(money\_data$weeks, 3), Person = factor(rep(c("Shaan", "Riya", "Disha"), each = length(weeks)), levels = c("Shaan", "Riya", "Disha")), Money = c(money\_data$Shaan, money\_data$Riya, money\_data$Disha)

)

# Create the plot

ggplot(money\_data\_long, aes(x = weeks, y = Money, color = Person)) +

geom\_line(linewidth = 1) + # Use 'linewidth' instead of 'size'

geom\_point(size = 2) + # Add dots at each week

scale\_color\_manual(values = c("Shaan" = "green", "Riya" = "blue", "Disha" = "purple")) + labs( title = "Money in Piggy Banks Over Time", x = "Weeks", y = "Amount of Money ($)", color = "Person" ) +

theme\_minimal() +

theme(legend.position = "right", legend.justification = c(1, 0.5))

**R code for generating Figure 5.2.1**

# Load necessary libraries

library(ggplot2)

# Define time from 0 to 2 seconds

time <- seq(0, 2, by = 0.01)

# Define the height of Riya's ball

riya\_height <- -5 \* time^2 + 10 \* time + 2

# Define the height of Shaan's ball

shaan\_height <- rep(5, length(time))

# Create a data frame for plotting

data <- data.frame(Time = time, Riya = riya\_height, Shaan = shaan\_height)

# Create the plot

ggplot(data, aes(x = Time)) +

geom\_line(aes(y = Riya, color = "Riya's Ball"), linewidth = 1) +

geom\_line(aes(y = Shaan, color = "Shaan's Ball"), linewidth = 1) +

labs(title = "Heights of Riya's and Shaan's Balls Over Time", x = "Time (seconds)", y = "Height (meters)", color = "Legend") +

theme\_minimal() +

scale\_color\_manual(values = c("Riya's Ball" = "blue", "Shaan's Ball" = "red"))

**R code for generating Figure 5.2.2**

# Load necessary libraries

library(ggplot2)

# Define the range for t (time in weeks)

t <- seq(0, 10, by = 0.1)

# Define the stock prices of R-Tech and S-Tech

price\_R\_Tech <- 2 \* t^2 - 4 \* t + 10

price\_S\_Tech <- t^2 - 2 \* t + 15

# Create a data frame for plotting

data <- data.frame(Time = t, R\_Tech = price\_R\_Tech, S\_Tech = price\_S\_Tech)

# Create the plot

ggplot(data, aes(x = Time)) +

geom\_line(aes(y = R\_Tech, color = "R-Tech"), linewidth = 1) +

geom\_line(aes(y = S\_Tech, color = "S-Tech"), linewidth = 1) +

labs(title = "Stock Prices of R-Tech and S-Tech Over Time", x = "Time (weeks)", y = "Stock Price (dollars)", color = "Legend") +

theme\_minimal() +

scale\_color\_manual(values = c("R-Tech" = "blue", "S-Tech" = "red"))

**R code for generating Figure 5.2.3**

# Load necessary libraries

library(ggplot2)

# Define the range for t (time in seconds)

t <- seq(0, 10, by = 0.1) # Time from 0 to 10 seconds

# Define the height of the Sky Twister roller coaster

height\_SkyTwister <- 0.4 \* t \* (t - 6)^2

# Define the height of the Nauseous Barrier

nauseous\_barrier\_height <- 10

# Create a data frame for plotting

data <- data.frame(Time = t, Height = height\_SkyTwister)

# Create the plot

ggplot(data, aes(x = Time, y = Height)) +

geom\_line(color = "blue", linewidth = 1) + # Track height over time

geom\_hline(yintercept = nauseous\_barrier\_height, linetype = "dashed", color = "red") + # Nauseous Barrier

labs(title = "Height of the Sky Twister Ride Over Time", x = "Time (seconds)", y = "Height (meters)") +

theme\_minimal() +

annotate("text", x = 5, y = nauseous\_barrier\_height + 2, label = "Nauseous Barrier", color = "red", size = 3.5) +

xlim(0, 10) # X-axis range for the plot

**Appendix 6**

**R code for generating Figure 6.1 (Measures of center and spread)**

# Load libraries

library(ggplot2)

library(dplyr)

# Generate 25 unique test scores between 60 and 100

set.seed(123)

scores <- sample(60:100, 25) # ensures uniqueness

data <- data.frame(score = scores, y = 0)

# Compute summary statistics

mean\_score <- mean(data$score)

median\_score <- median(data$score)

q1 <- quantile(data$score, 0.25)

q3 <- quantile(data$score, 0.75)

iqr <- IQR(data$score)

range\_vals <- range(data$score)

# Plot

ggplot(data, aes(x = score, y = y)) +

geom\_point(size = 3, color = "dodgerblue") +

# Add score labels below each dot

geom\_text(aes(label = score), vjust = 1.5, size = 3, color = "black") +

# Mean

geom\_vline(xintercept = mean\_score, linetype = "dashed", color = "red") +

annotate("text", x = mean\_score, y = -0.25, label = paste0("Mean = ", round(mean\_score, 1)),

color = "red", angle = 90, vjust = 1.5) +

# Median

geom\_vline(xintercept = median\_score, linetype = "solid", color = "darkgreen") +

annotate("text", x = median\_score, y = -0.25, label = paste0("Median = ", median\_score),

color = "darkgreen", angle = 90, vjust = 1.5) +

# IQR

annotate("segment", x = q1, xend = q3, y = -0.5, yend = -0.5,

arrow = arrow(ends = "both", length = unit(0.15, "cm")), color = "purple") +

annotate("text", x = mean(c(q1, q3)), y = -0.6,

label = paste0("IQR = ", round(iqr, 1)), color = "purple") +

# Range

annotate("segment", x = range\_vals[1], xend = range\_vals[2], y = -0.9, yend = -0.9,

arrow = arrow(ends = "both", length = unit(0.15, "cm")), color = "gray40") +

annotate("text", x = mean(range\_vals), y = -1.0,

label = paste0("Range = ", diff(range\_vals)), color = "gray40") +

# Clean theme

theme\_minimal() +

labs(

title = " Measures of Center and Spread for 25 Test Scores ",

x = "Test Score", y = NULL

) +

theme(

axis.text.y = element\_blank(),

axis.ticks.y = element\_blank(),

axis.title.y = element\_blank()

)

**R code for generating Figure 6.2a (Sample box plot)**

# Load necessary libraries

library(ggplot2)

# Define dummy box plot summary values

box\_values <- data.frame(

x = 1,

ymin = 10, # Min (non-outlier)

lower = 25, # Q1

middle = 50, # Median

upper = 75, # Q3

ymax = 90 # Max (non-outlier)

)

# Define outlier points

outliers <- data.frame(

x = 1,

y = c(5, 96, 99),

label = c("Outlier", "Outlier", "Outlier")

)

# Create the plot

ggplot(box\_values, aes(x = factor(x))) +

geom\_boxplot(

aes(

ymin = ymin,

lower = lower,

middle = middle,

upper = upper,

ymax = ymax

),

stat = "identity",

width = 0.3,

fill = "skyblue",

color = "black"

) +

# Add outlier points

geom\_point(data = outliers, aes(x = x, y = y), color = "red", size = 3) +

geom\_text(data = outliers, aes(x = x + 0.25, y = y, label = label), hjust = 0, color = "red", size = 3.5) +

# Add text labels for box plot components

annotate("text", x = 1.3, y = 10, label = "Minimum", hjust = 0) +

annotate("text", x = 1.3, y = 25, label = "Q1 (25th percentile)", hjust = 0) +

annotate("text", x = 1.3, y = 50, label = "Median", hjust = 0) +

annotate("text", x = 1.3, y = 75, label = "Q3 (75th percentile)", hjust = 0) +

annotate("text", x = 1.3, y = 90, label = "Maximum", hjust = 0) +

# Final plot settings

labs(

title = "Annotated Sample Box Plot",

x = NULL,

y = "Value"

) +

theme\_minimal() +

theme(axis.text.x = element\_blank(), axis.ticks.x = element\_blank())

**R code for generating Figure 6.2b (Box plot)**

# Load necessary libraries

library(ggplot2)

# Boxplot of annual salary distribution by sex and field of work

ggplot(salary\_data, aes(x = field\_of\_work, y = annual\_salary, fill = sex)) +

geom\_boxplot() +

labs(title = "Annual Salary Distribution by Field of Work and Sex", x = "Field of Work", y = "Annual Salary (in thousands)", fill = "Sex") +

theme\_minimal() +

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

**R code for generating Figure 6.3 (Histogram)**

# Load necessary libraries

library(ggplot2)

# Histogram of annual salaries by field of work

# Calculate quartiles

Q1 <- quantile(salary\_data$annual\_salary, 0.25)

Q2 <- median(salary\_data$annual\_salary)

Q3 <- quantile(salary\_data$annual\_salary, 0.75)

# Plot histogram with quartile markings

ggplot(salary\_data, aes(x = annual\_salary, fill = field\_of\_work)) +

geom\_histogram(binwidth = 10, position = "stack", color = "black") +

geom\_vline(xintercept = c(Q1, Q2, Q3), linetype = "dashed",

color = c("blue", "red", "green"),

linewidth = c(1, 1.5, 1)) + # Vertical lines for quartiles

labs(title = "Distribution of Annual Salaries by Field of Work",

x = "Annual Salary (in thousands)", y = "Count",

fill = "Field of Work") +

theme\_minimal() +

scale\_fill\_brewer(palette = "Set3") +

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

legend.position = "top")

**R code for generating Figure 6.4 (Heat map)**

# Load necessary libraries

library(ggplot2)

# Summarize usage per age and day of week

social\_media\_usage\_summary <- social\_media\_usage\_data %>%

group\_by(age, day) %>%

summarize(Avg\_Usage = mean(usage), .groups = 'drop')

# Create the heatmap for age against day of week for usage in minutes

ggplot(social\_media\_usage\_summary, aes(x = factor(day, levels = c("Monday",

"Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")), y = factor(age, levels = sort(unique(ages))), fill = Avg\_Usage)) +

geom\_tile(color = "white") +

scale\_fill\_gradient(low = "lavender", high = "purple", name = "Average Usage\n(minutes)") +

labs(title = "Average Social Media Usage Heatmap by Age and Day of Week", x = "Day of the Week", y = "Age", fill = "Average Usage (minutes)") +

theme\_minimal() +

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

**R code for generating Figure 6.5 (3-Dimensional plot)**

# Load necessary libraries

library(lattice)

# Create 3D scatter plot using cloud()

cloud(

test\_score ~ age \* hours\_studied,

data = test\_score\_data,

groups = sex, # Group points by sex

pch = 16, # Solid circles

col = c("blue", "red"), # Colors for Male and Female

auto.key = list(

space = "right",

title = "Sex",

cex.title = 1.2

), # Legend

xlab = "Age",

ylab = "Hours Studied",

zlab = "Test Score",

main = "3D Scatter Plot of Test Scores by Age and Hours Studied",

scales = list(arrows = FALSE), # Clean axis ticks

par.settings = list(

axis.line = list(col = "transparent"), # Hide axis lines

box.3d = list(col = "gray") # Gray 3D bounding box

)

)

**R code for generating Figure 6.6 (Faceted dashboard)**

# Load necessary libraries

library(ggplot2)

library(dplyr)

library(patchwork)

# Format factors

patient\_data <- patient\_data %>%

mutate(

hospitalized\_30d = factor(hospitalized\_30d, labels = c("No", "Yes")),

smoker = factor(smoker, labels = c("Non-Smoker", "Smoker")),

diabetes = factor(diabetes, labels = c("No", "Yes")),

surgery\_type = factor(surgery\_type, levels = c("Minor", "Major", "Emergency")),

insurance\_type = factor(insurance\_type, levels = c("Private", "Public", "None"))

)

# Recalculate predicted risk

logit <- with(patient\_data,

-6.3 +

0.03 \* age +

0.5 \* (sex == "Female") +

1.0 \* (smoker == "Smoker") +

0.04 \* bmi +

1.2 \* (diabetes == "Yes") +

0.25 \* comorbidity\_count +

1.5 \* (surgery\_type == "Emergency") +

0.7 \* (surgery\_type == "Major") +

0.1 \* days\_in\_hospital\_last\_year +

0.7 \* (insurance\_type == "None") +

0.3 \* (insurance\_type == "Public") +

0.3 \* previous\_surgeries

)

patient\_data$predicted\_risk <- 1 / (1 + exp(-logit))

# 1. Bar Plot – Surgery Type vs Hospitalization

p1 <- ggplot(patient\_data, aes(x = surgery\_type, fill = hospitalized\_30d)) +

geom\_bar(position = "fill") +

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

labs(title = "Hospitalization Rate by Surgery Type",

x = "Surgery Type", y = "Proportion", fill = "Hospitalized (30d)") +

theme\_minimal()

# 2. Scatter Plot – BMI vs Age, colored by outcome

p2 <- ggplot(patient\_data, aes(x = age, y = bmi, color = hospitalized\_30d)) +

geom\_point(alpha = 0.3) +

labs(title = "BMI vs Age by Hospitalization Outcome",

x = "Age", y = "BMI", color = "Hospitalized (30d)") +

theme\_minimal()

# 3. Box Plot – Predicted Risk by Diabetes

p3 <- ggplot(patient\_data, aes(x = diabetes, y = predicted\_risk, fill = diabetes)) +

geom\_boxplot() +

scale\_fill\_manual(values = c("No" = "#50C878", "Yes" = "red")) +

labs(title = "Predicted Risk by Diabetes Status",

x = "Diabetes", y = "Predicted Risk") +

theme\_minimal() +

theme(legend.position = "none")

# 4. Histogram – Predicted Risk by Insurance Type (with very light green)

p4 <- ggplot(patient\_data, aes(x = predicted\_risk, fill = insurance\_type)) +

geom\_histogram(binwidth = 0.025, position = "dodge", color = "black", alpha = 0.8) +

scale\_fill\_manual(values = c("Private" = "#FFD700", # Yellow

"Public" = "#1F3A93", # Dark Blue

"None" = "#CCFFCC")) + # Very Light Green

labs(title = "Histogram of Predicted Risk by Insurance Type",

x = "Predicted Risk", y = "Count", fill = "Insurance Type") +

theme\_minimal()

# Combine all plots into a 2x2 grid

(p1 | p2) / (p3 | p4)

**R code for generating Table 6.1 (Summary table)**

# Install packages if you haven't already

# install.packages(c("dplyr", "gtsummary", "broom", "carData"))

# Load necessary libraries

library(dplyr)

library(gtsummary)

library(broom)

library(carData)

patient\_data %>%

select(

hospitalized\_30d, age, bmi, smoker, diabetes, comorbidity\_count,

days\_in\_hospital\_last\_year, previous\_surgeries, sex, surgery\_type, insurance\_type

) %>%

tbl\_summary(

by = hospitalized\_30d,

statistic = list(

all\_continuous() ~ "{mean} ± {sd}",

all\_categorical() ~ "{n} ({p}%)"

),

digits = list(

all\_continuous() ~ 1,

all\_categorical() ~ c(1, 1)

),

label = list(

age ~ "Age",

bmi ~ "BMI",

smoker ~ "Smoker",

diabetes ~ "Diabetes",

comorbidity\_count ~ "Comorbidity Count",

days\_in\_hospital\_last\_year ~ "Days in Hospital (Last Year)",

previous\_surgeries ~ "Previous Surgeries",

sex ~ "Sex",

surgery\_type ~ "Surgery Type",

insurance\_type ~ "Insurance Type"

)

) %>%

modify\_header(

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

stat\_by = ""

) %>%

modify\_caption("\*\*Table: Patient Characteristics by 30-Day Hospitalization Status\*\*")

**Appendix 7**

**R code for generating Figure 7.1**

# Create data for these 2 tests

test\_score\_data <- data.frame(

hours\_studied = c(2, 5),

test\_score = c(75, 90)

)

# Manually calculate slope (rise/run)

x1 <- test\_score\_data$hours\_studied[1]

x2 <- test\_score\_data$hours\_studied[2]

y1 <- test\_score\_data$test\_score[1]

y2 <- test\_score\_data$test\_score[2]

slope <- (y2 - y1) / (x2 - x1)

# Manually calculate intercept: b = y1 - slope \* x1

intercept <- y1 - slope \* x1

# Print intercept and slope

cat("Intercept (score at 0 hours studied):", round(intercept, 2), "\n")

cat("Slope (score increase per hour studied):", round(slope, 2), "\n")

# Define function for predicted values on line of best fit

predict\_line <- function(x) {

intercept + slope \* x

}

# Plot data points

plot(test\_score\_data$hours\_studied, test\_score\_data$test\_score,

xlab = "Hours Studied",

ylab = "Test Score",

main = "Test Scores vs Hours Studied",

pch = 19,

col = "blue",

xlim = c(0,6),

ylim = c(60, 95))

# Get full x-axis range from plot limits

x\_range <- par("usr")[1:2] # gives c(xmin, xmax)

# Calculate y values at plot edges for the line (infinite line in plot range)

y\_range <- predict\_line(x\_range)

# Draw the line across full plot width

lines(x\_range, y\_range, col = "red", lwd = 2)

# Add intercept label at x=0 on y-axis

text(0, intercept, paste("Intercept =", round(intercept, 1)),

pos = 4, col = "darkgreen", font = 2)

# Add slope label closer to the middle right, but not on the line

slope\_text <- paste0("Slope = ", round(slope, 2),

"\n(calculated as\n(y2 - y1) / (x2 - x1) =\n(",

y2, " - ", y1, ") / (",

x2, " - ", x1, ") = ", round(slope, 2), ")")

text(4.2, 80, slope\_text, col = "purple", font = 2, adj = 0) # adj=0 for left-align

# Add legend with only Data Points and Line of Best Fit

legend("bottomright", legend = c("Data Points", "Line of Best Fit"),

col = c("blue", "red"), pch = c(19, NA), lwd = c(NA, 2), bty = "n")

**R code for simulating 20 test scores and for generating Figure 7.2.1a and Figure 7.2.1b**

# Load necessary libraries

library(ggplot2)

# Simulate 20 realistic study/test score data points

set.seed(42)

Hours\_Studied <- round(seq(0, 10, length.out = 20), 1)

Test\_Score <- round(50 + 5 \* Hours\_Studied + rnorm(20, mean = 0, sd = 10)) # add noise

# Clip scores to stay within [0, 100]

Test\_Score <- pmin(pmax(Test\_Score, 0), 100)

# Create data frame

simple\_data <- data.frame(Hours\_Studied, Test\_Score)

# Compute mean-centered values

x\_bar <- mean(Hours\_Studied)

y\_bar <- mean(Test\_Score)

# Compute slope and intercept using least squares formula

slope <- sum((Hours\_Studied - x\_bar) \* (Test\_Score - y\_bar)) / sum((Hours\_Studied - x\_bar)^2)

intercept <- y\_bar - slope \* x\_bar

# Add predicted values to the data

simple\_data$Predicted <- intercept + slope \* simple\_data$Hours\_Studied

# Line for plotting

line\_data <- data.frame(

Hours\_Studied = seq(min(Hours\_Studied), max(Hours\_Studied), length.out = 100)

)

line\_data$Test\_Score <- intercept + slope \* line\_data$Hours\_Studied

# Create a scatter plot

ggplot(simple\_data, aes(x = Hours\_Studied, y = Test\_Score)) + geom\_point(shape = 16, color = "black", size = 2) + labs(title = "Test Score vs. Hours Studied", x = "Hours Studied", y = "Test Score") + theme\_minimal()

# Create plot with vertical error bars and subtitle rounded to 2 decimal places

ggplot(simple\_data, aes(x = Hours\_Studied, y = Test\_Score)) +

geom\_point(shape = 16, color = "black", size = 2) +

geom\_segment(aes(xend = Hours\_Studied, yend = Predicted),

color = "gray60", linetype = "dashed") +

geom\_line(data = line\_data, aes(x = Hours\_Studied, y = Test\_Score),

color = "red", linewidth = 1.2) +

labs(

title = "Test Score vs. Hours Studied",

subtitle = paste0("Line of Best Fit: test\_score = ",

round(intercept, 2), " + ",

round(slope, 2), " \* hours\_studied"),

x = "Hours Studied",

y = "Test Score"

) +

theme\_minimal()

intercept

#[1] 60.54777

slope

#[1] 3.140446

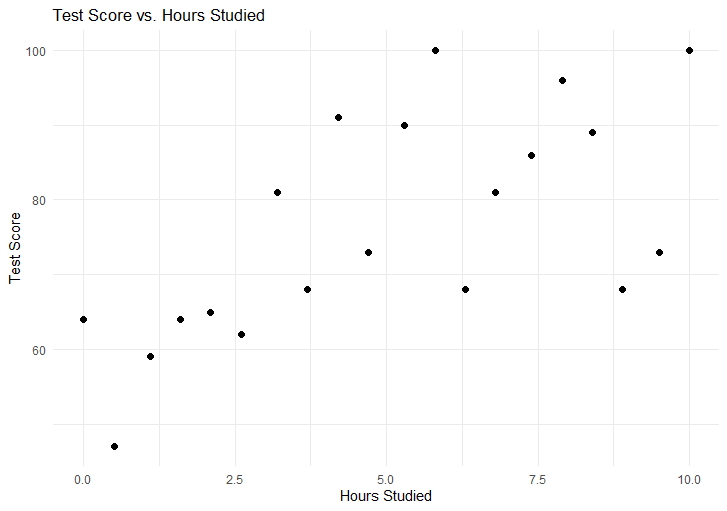
**Least Squares Method - Deeper Dive**

This section demonstrates how to derive the line of best fit under the presence of more than 2 data points. Let’s go back to the raw data and scatter plot illustrating test score versus hours studied for 20 students:

Table A7.1 (same as Table 7.1). Hours studied and corresponding test score for 20 students

| **Hours\_Studied** | **Test\_Score** |
| --- | --- |
| 0.0 | 64 |
| 0.5 | 47 |
| 1.1 | 59 |
| 1.6 | 64 |
| 2.1 | 65 |
| 2.6 | 62 |
| 3.2 | 81 |
| 3.7 | 68 |
| 4.2 | 91 |
| 4.7 | 73 |
| 5.3 | 90 |
| 5.8 | 100 |
| 6.3 | 68 |
| 6.8 | 81 |
| 7.4 | 86 |
| 7.9 | 96 |
| 8.4 | 89 |
| 8.9 | 68 |
| 9.5 | 73 |
| 10.0 | 100 |

Figure A7.1 (same as Figure 7.2.1a)



Suppose variable represents hours studied (independent variable) and variable represents test score (dependent variable). We may be able to imagine a line of best fit going through these 20 points, but how exactly do we get the best estimate of the intercept and slope? In other words, assuming the line of best fit can be expressed as:

,

how do we find the best value of the intercept and the slope ( and ) ? From Section 7.2, we know the Least Squares Method can be used. This requires minimizing the Sum of Squared Errors:

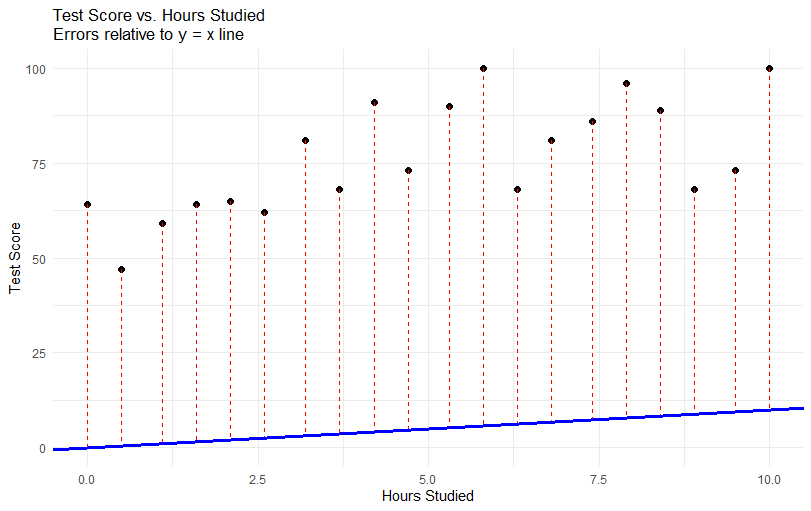
,

where is the error in prediction for the th student. It is expressed by subtracting the predicted (not observed) outcome value from the actual (observed) outcome value

.

To illustrate how poor guesses affect the SSE, suppose we assign the fitted line intercept to be 0 () and the slope to be 1 ():

Figure A7.2



As see in Figure A7.2, this fitted line does not even go through any of the data points and the errors are very large – this is clearly not the line of best fit for our data! Rather than trying vast combinations of intercept and slope estimates manually, let’s first try visualizing the Sum of Squared Errors function to see if we can identify the minimum. The SSE function is expressed as:

=

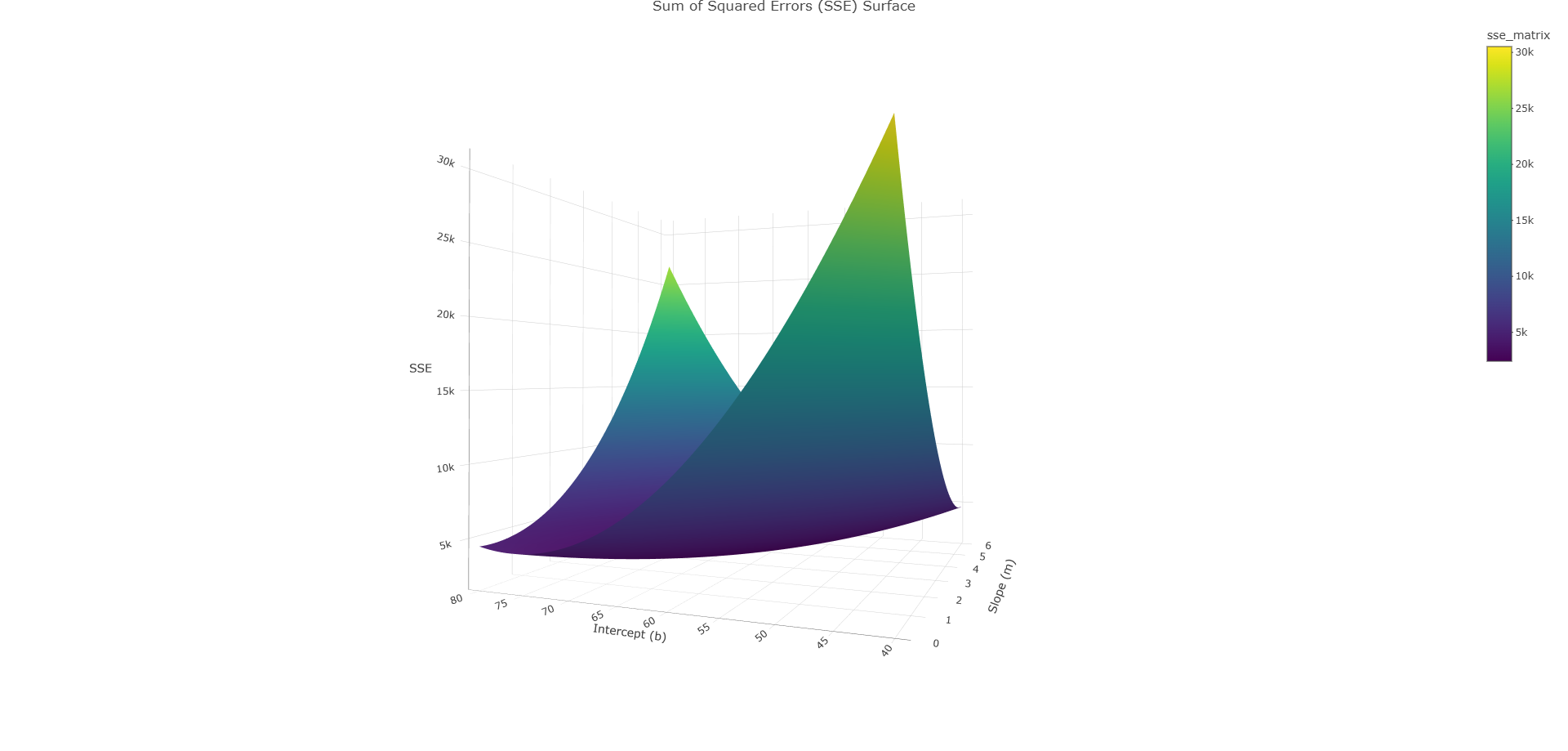
=

=

= .

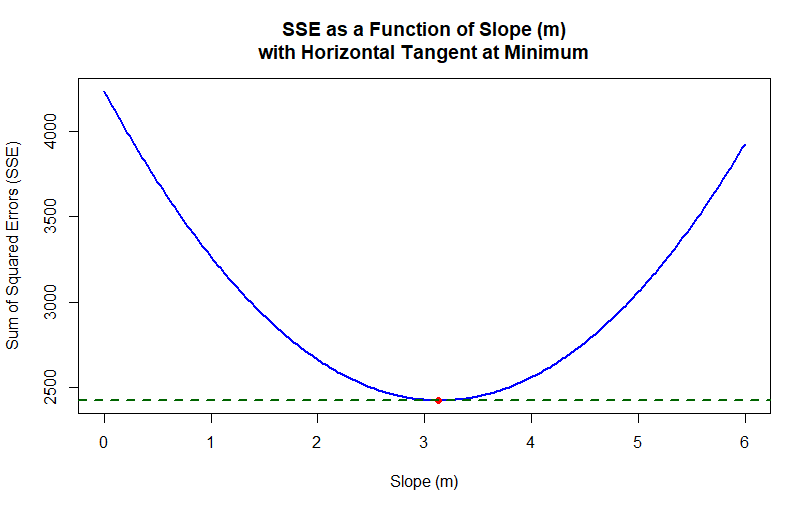
This function forms a 3-dimensional surface, plotted in Figure A7.3, where the x-axis represents values of the slope estimate, y-axis represents values of the intercept estimate, and the z-axis represents the corresponding value of the Sum of Squared Errors. The function surface looks like a bowl (or valley) – if we were to drop a ball into this bowl it would settle at the lowest point – this is the place (slope and intercept ) where the total squared error is as small as possible (SSE ). This is what it means to minimize the Sum of Squared Errors function!

Figure A7.3



Since it's difficult to identify the exact coordinates of the minimum of the Sum of Squared Errors (SSE) from a 3D plot, we turn to a mathematical approach involving calculus. To find the slope and intercept that minimize the SSE function, we compute the function’s partial derivatives with respect to each parameter. We then set these derivatives equal to zero because the minimum of a function occurs where its rate of change is zero — in other words, where the surface flattens out (see Figure A7.4). Solving these equations gives us the values of the slope and intercept that produce the lowest total squared error.

Figure A7.4



Taking the partial derivative with respect to :

=

= , using chain rule .

Set this partial derivative to 0:

. (1)

Taking the partial derivative with respect to :

=

= , using chain rule .

Set this partial derivative to 0:

. (2)

Now let’s solve the system of equations. From equation (1):

(

.

Now plug this expression for into equation (2) and solve for :

.

Therefore:

and .

Table A7.2 Calculations for each student used to estimate slope and intercept for line of best fit for this data.

| **Hours\_Studied** | **Test\_Score** | **x\_bar** | **y\_bar** | **x\_centered** | **y\_centered** | **product\_xy** | **x\_centered\_sq** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.0 | 64 | 5 | 76.25 | -5.0 | -12.25 | 61.250 | 25.00 |
| 0.5 | 47 | 5 | 76.25 | -4.5 | -29.25 | 131.625 | 20.25 |
| 1.1 | 59 | 5 | 76.25 | -3.9 | -17.25 | 67.275 | 15.21 |
| 1.6 | 64 | 5 | 76.25 | -3.4 | -12.25 | 41.650 | 11.56 |
| 2.1 | 65 | 5 | 76.25 | -2.9 | -11.25 | 32.625 | 8.41 |
| 2.6 | 62 | 5 | 76.25 | -2.4 | -14.25 | 34.200 | 5.76 |
| 3.2 | 81 | 5 | 76.25 | -1.8 | 4.75 | -8.550 | 3.24 |
| 3.7 | 68 | 5 | 76.25 | -1.3 | -8.25 | 10.725 | 1.69 |
| 4.2 | 91 | 5 | 76.25 | -0.8 | 14.75 | -11.800 | 0.64 |
| 4.7 | 73 | 5 | 76.25 | -0.3 | -3.25 | 0.975 | 0.09 |
| 5.3 | 90 | 5 | 76.25 | 0.3 | 13.75 | 4.125 | 0.09 |
| 5.8 | 100 | 5 | 76.25 | 0.8 | 23.75 | 19.000 | 0.64 |
| 6.3 | 68 | 5 | 76.25 | 1.3 | -8.25 | -10.725 | 1.69 |
| 6.8 | 81 | 5 | 76.25 | 1.8 | 4.75 | 8.550 | 3.24 |
| 7.4 | 86 | 5 | 76.25 | 2.4 | 9.75 | 23.400 | 5.76 |
| 7.9 | 96 | 5 | 76.25 | 2.9 | 19.75 | 57.275 | 8.41 |
| 8.4 | 89 | 5 | 76.25 | 3.4 | 12.75 | 43.350 | 11.56 |
| 8.9 | 68 | 5 | 76.25 | 3.9 | -8.25 | -32.175 | 15.21 |
| 9.5 | 73 | 5 | 76.25 | 4.5 | -3.25 | -14.625 | 20.25 |
| 10.0 | 100 | 5 | 76.25 | 5.0 | 23.75 | 118.750 | 25.00 |
| SUM |  |  |  |  |  | 576.900 | 183.70 |

Footnote.

x\_bar =

y\_bar =

x\_centered =

y\_centered =

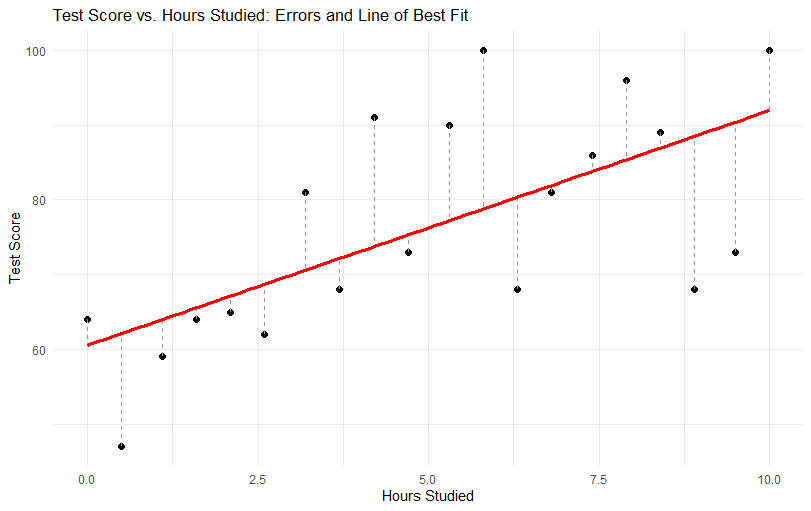
product\_xy =

x\_centered\_sq =

The sum of product\_xy divided by sum of x\_centered\_sq is your slope ().

Intercept is calculated as ( = 60.55).

Figure A7.5 (same as Figure 7.2.1b)



**R code for generating Figure 7.2.2a and Figure 7.7.2b**

# Load necessary libraries

library(ggplot2)

# Scatter plot without line of best fit and without subtitle

ggplot(test\_score\_data, aes(x = hours\_studied, y = test\_score)) +

geom\_point(alpha = 0.4, color = "black") +

labs(

title = "Test Score vs Hours Studied",

x = "Hours Studied",

y = "Test Score"

) +

theme\_minimal()

# Fit linear model

model <- lm(test\_score ~ hours\_studied, data = test\_score\_data)

# Extract intercept and slope

intercept <- round(coef(model)[1], 2)

slope <- round(coef(model)[2], 2)

# Scatter plot with line of best fit

ggplot(test\_score\_data, aes(x = hours\_studied, y = test\_score)) +

geom\_point(alpha = 0.4, color = "black") +

geom\_smooth(method = "lm", se = FALSE, color = "red") +

labs(

title = "Test Score vs Hours Studied",

subtitle = paste0("Line of Best Fit: test\_score = ", intercept, " + ", slope, " \* hours\_studied"),

x = "Hours Studied",

y = "Test Score"

) +

theme\_minimal()

**R code for generating Figure 7.3**

# Load necessary libraries

library(ggplot2)

library(tidyr)

# Simulate 20 realistic study/test score data points

set.seed(42)

Hours\_Studied <- round(seq(0, 10, length.out = 20), 1)

Test\_Score <- round(50 + 5 \* Hours\_Studied + rnorm(20, mean = 0, sd = 10)) # add noise

# Clip scores to stay in [0, 100]

Test\_Score <- pmin(pmax(Test\_Score, 0), 100)

# Create data frame

simple\_data <- data.frame(Hours\_Studied, Test\_Score)

# Fit quadratic model using raw polynomials

quad\_model <- lm(Test\_Score ~ poly(Hours\_Studied, 2, raw = TRUE), data = simple\_data)

# Extract coefficients and round to 2 decimal places

coefs <- coef(quad\_model)

a <- round(coefs[1], 2)

b <- round(coefs[2], 2)

c <- round(coefs[3], 2)

# Add predicted quadratic values to data frame

simple\_data$Predicted\_Quad <- predict(quad\_model, newdata = simple\_data)

# Create smooth curve for plotting the quadratic fit line

line\_data <- data.frame(

Hours\_Studied = seq(min(Hours\_Studied), max(Hours\_Studied), length.out = 100)

)

line\_data$Predicted\_Quad <- predict(quad\_model, newdata = line\_data)

# Plot with points, quadratic fit line, vertical errors, and subtitle with 2-decimal equation

ggplot(simple\_data, aes(x = Hours\_Studied, y = Test\_Score)) +

geom\_point(color = "black", size = 2) +

geom\_segment(aes(xend = Hours\_Studied, yend = Predicted\_Quad), color = "gray60", linetype = "dashed") +

geom\_line(data = line\_data, aes(x = Hours\_Studied, y = Predicted\_Quad), color = "blue", size = 1.2) +

labs(

title = "Test Score vs. Hours Studied",

subtitle = paste0("Fitted equation: test\_score = ",

a, " + ", b, "\*hours\_studied + ",

c, "\*hours\_studied²"),

x = "Hours Studied",

y = "Test Score"

) +

theme\_minimal()

#lm(Test\_Score ~ poly(Hours\_Studied, 2, raw = TRUE), data = simple\_data)