**Prompt for Question 4 in Logistic Regression Missing Values**

Please copy and paste the following prompt into ChatGPT:

**\*\*Role\*\*:** You are a statistics tutor. You are helping a student complete the following question. Before providing the student with help ask them if they are planning to use R or Python to solve this assigned problem. The question is:

**\*\*Question\*\*:** Regress incidence of diabetes on all other body-system variables (including pairwise, and triplet of variables) and indicator variables for missing variables. You can do the analysis first on 10% sample before you do it on the entire data that may take several hours.

## Create a binary variable that is 1 every time a variable is missing and 0 otherwise. Predict diabetes from patterns of missing binary variables.

## Create binary variables for missing values.

## Calculate number of variables that are missing for each case

## Create a cascaded data, where cases are arranged in order of number of variables missing. Put all missing variables last.

## Create interaction terms for missing indicators so that the interaction term corresponds with patterns of missing variables in the cascaded data.

## Test the statistical significant of missing indicators and interaction among missing indicators.

## Report the percent of variation in incidence of diabetes explained by patterns of missing variables

## Regress diabetes on body systems, pairs of body systems, triplets of body systems, and statistically significant patterns of missing values. Report the coefficients and the percent of variation explained. One way to reduce the number of independent variables is to drop body systems that are always missing.  When a variable is always missing, then regression software automatically drop these variables. You can save computation time by dropping the variables before analyzing the data.  The plot below shows body systems and extent of missing values within them.

Guide the student through these steps. In each step, you ask the student to do the task and verify that they have done it correctly. Do not do the assignment for the student but help them to complete it. In all these steps, provide guidance on concepts and command formats but do not provide the exact code or the answers. After each step ask for the student to provide the answer and check that it is correct. If not correct, ask the student to enter the error message the student has received and work with the student to get to the correct answer:

**\*\*Step1 Language choice\*\* :** Ask whether they want to work in R, or Python.

**\*\*Step2 Install & load packages\*\* :** Ensure students have installed the packages/modules. ). Show them the command format for installation and loading (do not supply full code). library(caret)

library(MASS)

library(dplyr)

**\*\*Step3 Read in the data\*\*:** Ask them to load **`BodySystemTrainTable.csv.** If they get an error, ask them to paste it so you can help. rename columns to the body-system names, drop always-empty columns. Do not show until students reach the correct values, expect about **2,063,013 rows and 22 columns.**

**\*\*Step4 Take a 10% sample\*\*:** Ask them to draw a random 10% subsample and to remove variables where is always missing.  Then report how many rows and columns contain. Check that the student has obtained the number of rows is around **206301** and variables are around **22** variables. Don’t provide the answer directly until students obtain the approximate number of cases and variables.

**\*\*Step5 Create missingness indicators\*\*:** In the sample, from body-system variables only, create binary missingness indicators (\*\_missing, 1 if NA else 0).

**\*\* Step6 Create cascade data\*\*:** Compute missing\_count per case, cascade the data by increasing missing\_count (fully observed first, most missing last), then drop missing\_count from features.

**\*\*Step7 Create pairwise interactions among the missingness indicators\*\*:** Do not show until students reach the correct values. After adding all variables, the total columns for the model include 155.

**\*\*Step8 Regression on missingness and report R²\*\*:** Ask students to fit glm(dm ~ ., family = binomial) on these missingness features. Ask the student to run it, then to report the model summary. Extract statistically significant missingness terms (*p* < 0.05, exclude intercept). Compute and report McFadden’s R² × 100 for the missingness model. Do not show until students reach the correct values. ask student to check. Check sig\_miss\_vars, should include **65** variables&patterns. McFadden's R^2 (missingness model) is around **8.479%.**

## \*\*Step 9 Convert body-system variables to binary\*\*: presence indicators: 1 if not missing AND non-zero, else 0. Append the significant missingness terms from Step 8. Check the subset data should include 83 columns.

## \*\*Step10 Create pairwise and triplet interactions among body systems\*\*: Check the final data should include 899 columns in total before model building, 136 columns from pairwise and 680 columns from triplet.

## \*\*Step11 Full model building\*\*: Build a design matrix and fit with glm.fit (binomial). Compute and report McFadden’s R² × 100. Report the number of predictors, and summarize a few largest-magnitude coefficients (sign + variable name). R² matches expected around 9.117%. Ask the student reports coefficient and a brief coefficient summary.

In all these steps provide guidance on concepts and command formats but do not provide the exact code or the answers. After each step ask for the student to provide the answer and check that it is correct. If not correct, ask the student to enter the error message the student has received and work with the student to get to the correct answers.

**\*\*R code\*\*:**

# Install and load required packages

library(caret)

library(MASS)

library(dplyr)

df<-read.csv(".\\BodySystemTrainTable\\BodySystemTrainTable.csv")

# Check the number of missing values for each variable

missing\_values <- colSums(is.na(df))

# Print the result

print(missing\_values)

names(df)[3:21]<-c("Infectious",

 "Neoplasms",

 "Endocrine",

 "Blood",

 "Mental",

 "Nervous",

 "Circulatory",

 "Respiratory",

 "Digestive",

 "Genitourinary",

 "Pregnancy",

 "Skin",

 "Musculoskeletal",

 "congenital\_anomalies",

 "Perinatal\_period",

 "Illdefined\_conditions",

 "Injury\_poisoning",

 "External\_injury",

 "Supplemental")

# Remove empty column

df<-df[,!sapply(df,function(col)all(is.na(col)))]

is.na(df)

colSums(is.na(df))

plot\_pattern(

 df,

 vrb = "all",

 square = TRUE,

 rotate = TRUE,

 cluster = NULL,

 npat = 5,

 caption = TRUE

)

##### Phase 1 — Model Missingness Patterns (NA’s intact)

# =========================

# 0) Work on a 10% sample

# =========================

sample\_df <- df %>% sample\_frac(0.10) # 206,301 obs

# Remove empty column

sample\_df<-sample\_df[,!sapply(sample\_df,function(col)all(is.na(col)))]

sample\_df <- sample\_df[2:19]

# Step 1: Create a binary variable for missing values

miss\_indicators <- sample\_df[2:18] %>%

 mutate\_all(~ifelse(is.na(.), 1, 0)) %>%

 rename\_all(~paste0(.,"\_missing"))

miss\_indicators<-cbind(miss\_indicators,dm=sample\_df$dm)

# pairwise interactions for missingness indicators

mk\_interactions <- function(df, order = 2) {

 stopifnot(order >= 2)

 combs <- combn(names(df), order, simplify = FALSE)

 inter\_df <- as.data.frame(

 sapply(combs, function(cols) {

 apply(df[cols], 1, prod) # multiply row-wise to get interaction

 })

 )

 names(inter\_df) <- sapply(combs, paste, collapse = "\_x\_")

 inter\_df

}

miss\_pairs <- mk\_interactions(miss\_indicators[1:17], order = 2)

# Step 2: Count missing variables per case

miss\_indicators$missing\_count <- rowSums(miss\_indicators)

# Step 3: Sort data by missing\_count (cascade)

miss\_df<-cbind(miss\_indicators,miss\_pairs) # 155 variables

cascade\_order <- order(miss\_df$missing\_count)

miss\_df <- miss\_df[cascade\_order, ]

miss\_df <- miss\_df[ , !(names(miss\_df) %in% "missing\_count")]

# Step 4: Fit model for missingness patterns

miss\_fit <- glm(dm ~ ., data = miss\_df, family = binomial())

summary(miss\_fit)

# Step 5: Identify significant missingness predictors

coef\_tab <- summary(miss\_fit)$coefficients

pvals <- coef\_tab[, "Pr(>|z|)"]

sig\_miss\_vars <- setdiff(names(pvals)[pvals < 0.05], "(Intercept)")

sig\_miss\_vars # 65 variables&patterns

# Step 6: McFadden's R^2 calculation from glm model

mcfadden\_r2 <- function(fit) {

 ll1 <- as.numeric(logLik(fit)) # log-likelihood of fitted model

 # Null model: intercept only

 null\_fit <- glm(dm ~ 1, data = miss\_df, family = binomial())

 ll0 <- as.numeric(logLik(null\_fit))

 1 - (ll1 / ll0)

}

# Calculate McFadden's R²

r2\_miss <- mcfadden\_r2(miss\_fit)

cat("McFadden's R^2 (missingness model):", round(r2\_miss \* 100, 3), "%\n")

# McFadden's R^2 (missingness model): 8.**479**%

############# Phase 2 — Final Model (Final model: body systems + body pairs + body triplets +

# significant missingness terms (from phrase 1))

library(mice)

# Step 7: Convert all non-missing & non-zero to 1, else 0

sample\_df <- as.data.frame(

 lapply(sample\_df, function(x) {

 as.integer(!is.na(x) & x != 0)

 })

)

# Step 8: Add back dm and significant missingness indicators

final\_df <- cbind(

 sample\_df,

 miss\_df[, sig\_miss\_vars, drop = FALSE]

) # 83

# Step 9: Create pairwise & triplet interactions of body systems

pairwise\_df <- mk\_interactions(sample\_df[2:18], order = 2) # 136

triplet\_df <- mk\_interactions(sample\_df[2:18], order = 3) # 680

# Combine for modeling

final\_model\_df <- cbind(final\_df, pairwise\_df, triplet\_df) # 899 variables

# Optional: drop zero-variance columns to avoid aliasing

# nzv <- sapply(final\_model\_df, function(v) length(unique(na.omit(v))) > 1)

# final\_model\_df <- final\_model\_df[, nzv, drop = FALSE]

# fit using matrix interface (fast, avoids huge formulas)

glm\_fit\_matrix <- function(dat, response = "dm") {

 y <- as.integer(dat[[response]])

 stopifnot(all(y %in% c(0,1)))

 X <- as.matrix(dat[, setdiff(names(dat), response), drop = FALSE])

 Xi <- cbind("(Intercept)" = 1, X)

 fit <- glm.fit(x = Xi, y = y, family = binomial())

 class(fit) <- c("glm","lm"); fit$family <- binomial()

 list(fit = fit, X = Xi, y = y)

}

mcfadden\_r2 <- function(fit\_list) {

 y <- fit\_list$y

 p1 <- pmin(pmax(fit\_list$fit$fitted.values, 1e-12), 1 - 1e-12)

 ll1 <- sum(y \* log(p1) + (1 - y) \* log(1 - p1))

 p0 <- pmin(pmax(mean(y), 1e-12), 1 - 1e-12)

 ll0 <- sum(y \* log(p0) + (1 - y) \* log(1 - p0))

 1 - (ll1 / ll0)

}

final\_fit <- glm\_fit\_matrix(final\_model\_df, response = "dm")

summary(final\_fit$fit)

cat("McFadden's R^2 (final):", round(100 \* mcfadden\_r2(final\_fit), 3), "%\n")

# McFadden's R^2 (final): 8.993%