**Supplementary information – R code for the analysis**

install.packages("vip","readxl","pls","ggplot2","pROC","dplyr")

#### PLS-R analysis#############################################################################################################

# Load necessary libraries

library(pls)

library(readxl)

# Read your Excel file - replace 'your\_data.xlsx' with the path to your Excel file

# And replace 'Sheet1' with the name of the sheet that contains your data

data <- read\_excel("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/Raw data.xlsx", sheet = "Sheet1")

# Assuming 'CD8A' is the column you want to predict and it's located at the end of your dataframe

# Exclude the first five columns which are non-predictive and the 'CD8A' column

X <- data[, -c(1:6, which(names(data) %in% c("CD8A","CD3")))]

Y <- data$CD8A

# Fit the Partial Least Squares Regression model

pls.model <- plsr(Y ~ ., data = cbind(Y, X), ncomp = 10, validation = "CV")

# Summary of the model

summary(pls.model)

# For plots and diagnostics

plot(pls.model)

# For a fitting curve, you need to plot observed vs. predicted values

# Extract the predicted values from the model

predictions <- predict(pls.model, ncomp = 10)

# Extract the observed values (assuming 'response' is your response variable)

observed <- data$CD8A

shape\_var <- ifelse(data$Molecular\_Response == "Responder", 3, 1)

# Plot the points with different shapes

plot(observed, predictions, pch = shape\_var, xlab = "Observed Values", ylab = "Predicted Values", main = "GBM-Pembro->25% CD45+ cells ROIs")

# Add legend

legend("topleft", legend = c("Responder", "Non-responder"), pch = c(3, 1))

#fitting curve

abline(0, 1, col = "red")

# Assuming cross-validation has been performed on the `pls\_model`

validationplot(pls.model, val.type = "MSEP")

# Assuming `actual\_values` are your observed Y values and `pls\_model` is your fitted model.

# Replace `pls\_model` with the actual variable name of your fitted PLS model.

# Step 1: Extract predicted values from the cross-validation (assuming the model has them)

predicted\_values <- predict(pls.model, ncomp = 10, newdata = cbind(Y, X), validation = "CV")

# Step 2: Calculate PRESS

PRESS <- sum((observed - predictions)^2)

# Step 3: Calculate TSS

TSS <- sum((observed - mean(observed))^2)

# Step 4: Calculate Q^2

Q2 <- 1 - (PRESS/TSS)

# Print Q2 value

print(Q2)

######### PLS-R ----- VIP or Coefficient generation #################################################################################

install.packages("vip")

library(vip)

# Assuming your PLS model is named 'pls\_model'

vip\_scores <- vip(pls.model, method = "model", num\_features = ncol(pls.model$model))

Most\_important\_vip\_scores <- vip(pls.model, method = "model")

plot(vip\_scores)

print(vip\_scores)

# Assuming vip\_scores$data contains the numerical values of the VIP scores

vip\_data <- vip\_scores$data

# Now write the data frame to a CSV file

write.csv(vip\_data, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLS\_VIP.csv", row.names = TRUE)

coefficients <- coef(pls.model, ncomp = 10)

# Write the matrix to a CSV file.

write.csv(coefficients, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLS\_Coefficients.csv",row.names = TRUE)

##Combination of VIP and Coefficient######

# Load the necessary package for writing Excel files

install.packages("writexl")

library(writexl)

# Read the CSV files

vip\_file <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLS\_VIP.csv", stringsAsFactors = FALSE)

coef\_file <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLS\_Coefficients.csv", stringsAsFactors = FALSE)

# Merge the two data frames by the 'Gene' column

combined\_data <- merge(vip\_file, coef\_file, by = "Gene", all = TRUE)

# Check the first few rows of the combined data frame

head(combined\_data)

# Write the combined data frame to a new Excel file

write\_xlsx(combined\_data, "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/VIP-Coefficients.xlsx")

#Plot VIP-Coefficient##############################################################################################################################

library(ggrepel)

library(ggplot2)

# Create a new column to determine if the point is above the threshold

combined\_data$above\_threshold <- combined\_data$Importance > 0.04

ggplot(combined\_data) +

# Points below the threshold, no labels, default color

geom\_point(aes(x = Y.10.comps, y = Importance), color = "grey", data = subset(combined\_data, above\_threshold == FALSE)) +

# Points above the threshold with labels, red color

geom\_point(aes(x = Y.10.comps, y = Importance), color = "red", data = subset(combined\_data, above\_threshold == TRUE)) +

geom\_text\_repel(aes(x = Y.10.comps, y = Importance, label = ifelse(above\_threshold, as.character(Gene), "")),

box.padding = 0.35, point.padding = 0.5,

data = subset(combined\_data, above\_threshold == TRUE)) +

theme\_minimal() +

labs(x = "Coefficients", y = "VIP Scores", title = "VIP Scores vs. Coefficients") +

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

geom\_hline(yintercept = 0.04, linetype = "dashed", color = "blue") +

geom\_vline(xintercept = 0, linetype = "dashed", color = "black")+ # This is the y=0.04 line

theme(

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

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

panel.background = element\_blank(),

panel.border = element\_rect(colour = "black", fill=NA, size=1)

) # Ensure this parenthesis is closed before the library call

######### PLS-DA analysis ################################################################################################################

# Load necessary libraries

library(pls)

library(readxl)

# Read your Excel file - replace 'your\_data.xlsx' with the path to your Excel file

# And replace 'Sheet1' with the name of the sheet that contains your data

data <- read\_excel("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/Raw data\_plsda round 2.xlsx", sheet = "Sheet1")

# Assuming 'CD8A' is the column you want to predict and it's located at the end of your dataframe

# Exclude the first five columns which are non-predictive and the 'CD8A' column

X <- data[, -c(1:6)]

Y <- data$pls\_code

# Fit the Partial Least Squares Regression model

plsda.model <- plsr(Y ~ ., data = cbind(Y, X), ncomp = 3, validation = "CV")

# Summary of the model

summary(plsda.model)

# For plots and diagnostics

plot(plsda.model)

# For a fitting curve, you need to plot observed vs. predicted values

# Extract the predicted values from the model

predictions <- predict(plsda.model, ncomp = 3)

write.csv(predictions, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLSda\_round2\_predictions.csv",row.names = TRUE)

# Extract the observed values (assuming 'response' is your response variable)

observed <- data$pls\_code

shape\_var <- ifelse(data$Molecular\_Response == "Responder", 3, 1)

# Plot the points with different shapes

plot(observed, predictions, pch = shape\_var, xlab = "Observed Values", ylab = "Predicted Values", main = "GBM-Pembro->25% CD45+ cells ROIs")

# Add legend

legend("topleft", legend = c("Responder", "Non-responder"), pch = c(3, 1))

#fitting curve

abline(0, 1, col = "red")

# Assuming cross-validation has been performed on the `pls\_model`

validationplot(plsda.model, val.type = "MSEP")

# Assuming `actual\_values` are your observed Y values and `pls\_model` is your fitted model.

# Replace `pls\_model` with the actual variable name of your fitted PLS model.

# Step 1: Extract predicted values from the cross-validation (assuming the model has them)

predicted\_values <- predict(plsda.model, ncomp = 3, newdata = cbind(Y, X), validation = "CV")

# Step 2: Calculate PRESS

PRESS <- sum((observed - predictions)^2)

# Step 3: Calculate TSS

TSS <- sum((observed - mean(observed))^2)

# Step 4: Calculate Q^2

Q2 <- 1 - (PRESS/TSS)

# Print Q2 value

print(Q2)

#PLS-DA ---- VIP or Coeffi generation #######################################################################################################

install.packages("vip")

library(vip)

# Assuming your PLS model is named 'pls\_model'

vip\_scores <- vip(plsda.model, method = "model", num\_features = ncol(plsda.model$model))

Most\_important\_vip\_scores <- vip(plsda.model, method = "model")

plot(Most\_important\_vip\_scores)

plot(vip\_scores)

print(vip\_scores)

# Assuming vip\_scores$data contains the numerical values of the VIP scores

vip\_data <- vip\_scores$data

# Now write the data frame to a CSV file

write.csv(vip\_data, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLSda\_VIP\_round\_1.csv", row.names = TRUE)

coefficients <- coef(plsda.model, ncomp = 3)

# Write the matrix to a CSV file.

write.csv(coefficients, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLSda\_Coefficients\_round\_1.csv",row.names = TRUE)

##Combine VIP and Coeffi##########################################################################################################################

# Load the necessary package for writing Excel files

install.packages("writexl")

library(writexl)

# Read the CSV files

vip\_file <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLSda\_VIP\_round\_1.csv", stringsAsFactors = FALSE)

coef\_file <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/CD8A-PLSda\_Coefficients\_round\_1.csv", stringsAsFactors = FALSE)

# Merge the two data frames by the 'Gene' column

combined\_data\_plsda\_round\_1 <- merge(vip\_file, coef\_file, by = "Gene", all = TRUE)

# Check the first few rows of the combined data frame

head(combined\_data\_plsda\_round\_1)

# Write the combined data frame to a new Excel file

write\_xlsx(combined\_data\_plsda\_round\_1, "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/VIP-Coefficients\_plsda\_round\_1.xlsx")

#Plot VIP-Coeffi################################################################################################################################

library(ggrepel)

library(ggplot2)

# Create a new column to determine if the point is above the threshold

combined\_data\_plsda\_round\_1$above\_threshold <- combined\_data\_plsda\_round\_1$Importance > 0.05

ggplot(combined\_data\_plsda\_round\_1) +

# Points below the threshold, no labels, default color

geom\_point(aes(x = Y.3.comps, y = Importance), color = "grey", data = subset(combined\_data\_plsda\_round\_1, above\_threshold == FALSE)) +

# Points above the threshold with labels, red color

geom\_point(aes(x = Y.3.comps, y = Importance), color = "red", data = subset(combined\_data\_plsda\_round\_1, above\_threshold == TRUE)) +

geom\_text\_repel(aes(x = Y.3.comps, y = Importance, label = ifelse(above\_threshold, as.character(Gene), "")),

box.padding = 0.35, point.padding = 0.5,

data = subset(combined\_data\_plsda\_round\_1, above\_threshold == TRUE)) +

theme\_minimal() +

labs(x = "Coefficients", y = "VIP Scores", title = "VIP Scores vs. Coefficients") +

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

geom\_hline(yintercept = 0.08, linetype = "dashed", color = "blue") +

geom\_vline(xintercept = 0, linetype = "dashed", color = "black")+ # This is the y=0.04 line

theme(

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

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

panel.background = element\_blank(),

panel.border = element\_rect(colour = "black", fill=NA, size=1)

) # Ensure this parenthesis is closed before the library call

# PLS-DA ---- box-plot ##################################################################################################################

boxplot\_data <- read\_excel("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/Raw data\_plsda round 2.xlsx", sheet = "Sheet1")

library(dplyr)

library(ggplot2)

# Assuming your data frame is named 'boxplot\_data' and the predictions are in a column named 'Predictions'

# Group by 'Molecular\_Response' and calculate the min and max

data\_summary <- boxplot\_data %>%

group\_by(Molecular\_Response) %>%

summarise(

min\_value = min(Predictions),

max\_value = max(Predictions)

)

# Merge the summary back with the original data to get the min and max for each observation

boxplot\_data <- merge(boxplot\_data, data\_summary, by = "Molecular\_Response", all.x = TRUE)

# Create the plot

ggplot(boxplot\_data, aes(x = Molecular\_Response, y = Predictions, fill = Molecular\_Response)) +

geom\_boxplot(outlier.shape = NA) +

geom\_jitter(width = 0.2, color = "black", size = 1.5, alpha = 0.6) +

geom\_errorbar(aes(ymin = min\_value, ymax = max\_value), width = 0.2) +

labs(

title =

"GBM-Pembro->25% CD45+cells ROIs

Prediction = 0.04753499\*CD11c - 0.06960034\*CD163

- 0.19743177\*CD44 - 0.22050609\*CD66B

+ 0.17226385\*PTEN + 0.14502225",

x = "Molecular Response Group",

y = "Predictions"

) +

theme\_minimal() +

scale\_fill\_manual(values = c("Nonresponder" = "lightblue", "Responder" = "lightpink")) +

theme(

panel.grid.major = element\_blank(), # Remove major grid lines

panel.grid.minor = element\_blank(), # Remove minor grid lines

panel.border = element\_rect(colour = "black", fill = NA, size = 1), # Add an outline around the plot panel

panel.background = element\_blank(), # Make panel background transparent

axis.line = element\_line(color = "black") # Add axis lines

)

#PLS-DA -----box plot------ROC curve ############################################################################################################

install.packages("pROC")

library(pROC)

# Compute ROC curve

roc\_object <- roc(boxplot\_data$pls\_code, boxplot\_data$Predictions)

# Plot ROC curve with specified parameters

plot(roc\_object, main="ROC Curve", col="red", lwd=5, xaxs="i", yaxs="i", xlim=c(1, 0), ylim=c(0, 1))

# Calculate AUC and add it to the plot

auc\_value <- auc(roc\_object)

text(x=0.8, y=0.2, labels=paste("AUC =", round(auc\_value, 4)), adj=c(-2,5))

######### Box plot for Ki67 expression between mR and mNR #####################

library(ggplot2)

library(dplyr)

library(readxl)

# Read the data from the 'immune-low' sheet

data\_immune\_low <- read\_excel("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/Raw data.xlsx", sheet = "immune-low")

ggplot(data\_immune\_low, aes(x = Molecular\_Response, y = Ki67, fill = Molecular\_Response)) +

geom\_boxplot(outlier.shape = NA) +

geom\_jitter(width = 0.2, color = "black", size = 1.5, alpha = 0.6) +

geom\_errorbar(aes(ymin = min(Ki67), ymax = max(Ki67)), width = 0.2) +

labs(

title = "Ki67 expression",

x = "Molecular Response Group",

y = "Ki67 expression"

) +

theme\_minimal() +

scale\_fill\_manual(values = c("Nonresponder" = "lightblue", "Responder" = "lightpink")) +

theme(

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

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

panel.border = element\_rect(colour = "black", fill = NA, size = 1),

panel.background = element\_blank(),

axis.line = element\_line(color = "black")

)

##########scatter plot#############

library(ggplot2)

data\_immune <- read\_excel("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Nat commun data/Raw data.xlsx", sheet = "GBM")

# Ensure 'Molecular\_Response' and 'Ki67' are factors

data\_immune$Molecular\_Response <- as.factor(data\_immune$Molecular\_Response)

data\_immune$Ki67 <- as.numeric(as.character(data\_immune$Ki67)) # Assuming Ki67 was a factor and originally numeric

# Create the scatter plot

ggplot(data\_immune, aes(x = `IDO-1`, y = `B7-H3`, color = Ki67, shape = Molecular\_Response)) +

geom\_point() +

scale\_shape\_manual(values = c("Responder" = 16, "Nonresponder" = 17)) +

scale\_color\_gradient(low = "lightgrey", high = "black") + # Adjust the color gradient as needed

labs(title = "Gene Expression Comparison",

x = "IDO-1 Expression",

y = "B7-H3 Expression",

color = "Ki67 Expression Level",

shape = "Molecular Response") +

theme\_minimal()

##############volcano plot for cell distribution in micro-tissue################

# Read in the data

data\_immune\_high <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/absolute cell fraction.csv")

library(ggplot2)

library(dplyr)

# Separate the cell type columns (excluding non-cell type columns)

cell\_types <- data\_immune\_high[, !(colnames(data\_immune\_high) %in% c("Tissue", "ROI", "Mixture", "P-value", "Correlation", "RMSE", "Absolute score (sig.score)"))]

# Initialize a data frame to store the results

results <- data.frame(CellType = character(), MeanDifference = numeric(), LogPValue = numeric())

# Loop through each cell type

for (cell\_type in colnames(cell\_types)) {

# Perform Wilcoxon test

test\_result <- wilcox.test(formula = as.formula(paste(cell\_type, "~ Tissue")), data = data\_immune\_high)

# Calculate the mean difference

mean\_diff <- mean(data\_immune\_high[data\_immune\_high$Tissue == "GI", cell\_type]) -

mean(data\_immune\_high[data\_immune\_high$Tissue == "Lung", cell\_type])

# Calculate -log10 of the p-value

log\_pvalue <- -log10(test\_result$p.value)

# Add the results to the data frame

results <- rbind(results, data.frame(CellType = cell\_type, MeanDifference = mean\_diff, LogPValue = log\_pvalue))

}

# Create the volcano plot

ggplot(results, aes(x = MeanDifference, y = LogPValue)) +

geom\_point() +

labs(title = "Volcano Plot", x = "Mean Difference (GI - Lung)", y = "-log10(P-value)") +

theme\_minimal()

# Save the plot

ggsave("volcano\_plot.jpeg", width=10, height=6)

################bar plot for immune fraction####################

# Load necessary libraries

library(tidyverse)

library(ggplot2)

# Read the dataset

data <- read.csv('C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/absolute cell fraction.csv')

# Select the relevant columns (ROI and cell types)

data\_selected <- data %>%

select(Tissue, ROI, `B.cells.naive`, `B.cells.memory`, `Plasma.cells`,

`T.cells.CD8`, `T.cells.CD4.naive`, `T.cells.CD4.memory.resting`,

`T.cells.CD4.memory.activated`, `T.cells.follicular.helper`,

`T.cells.regulatory..Tregs.`, `T.cells.gamma.delta`,

`Dendritic.cells.resting`, `Dendritic.cells.activated`)

# Melt the data for plotting

data\_long <- data\_selected %>%

gather(key = "Cell\_Type", value = "Fraction", -Tissue, -ROI)

my\_colors <- c("#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd",

"#8c564b", "#e377c2", "#7f7f7f", "#bcbd22", "#17becf",

"#1a55FF", "#ffbb78", "#98df8a", "#ff9896", "#c5b0d5",

"#c49c94", "#f7b6d2", "#c7c7c7", "#dbdb8d", "#9edae5",

"#6b6ecf", "#b5cf6b", "#d6616b", "#ad494a", "#8c6d31")

ggplot(data\_long, aes(x = factor(ROI), y = Fraction, fill = Cell\_Type)) +

geom\_bar(stat = "identity") +

scale\_fill\_manual(values = my\_colors) +

facet\_wrap(~Tissue, scales = "free\_x") +

theme\_minimal() +

labs(x = "ROI #", y = "Fraction of Immune Cells", title = "Cell Distribution per ROI") +

theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1))

##########box plot for immune fraction#############

# Load necessary libraries

library(tidyverse)

library(ggplot2)

# Read the dataset

data <- read.csv('C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/absolute cell fraction.csv')

# Load necessary libraries

library(ggplot2)

library(readr)

# Load your data

data <- read\_csv("path\_to\_your\_csv/absolute cell fraction.csv")

# Create the box plot

ggplot(data, aes(x=Tissue, y=Macrophages.M1)) +

geom\_boxplot() +

labs(title="Box Plot of Macrophage M1 Fraction by Tissue Type",

x="Tissue Type",

y="Macrophage M1 Fraction")

# Save the plot

ggsave("macrophage\_m1\_fraction\_boxplot.jpeg", width=10, height=6)

#########PLS-R analysis on M1 macrophages###########

# Load necessary libraries

library(pls)

library(readxl)

# Read your Excel file - replace 'your\_data.xlsx' with the path to your Excel file

# And replace 'Sheet1' with the name of the sheet that contains your data

data <- read\_excel("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/whole cell fraction.xlsx")

# Assuming 'CD8A' is the column you want to predict and it's located at the end of your dataframe

# Exclude the first five columns which are non-predictive and the 'CD8A' column

X <- data[, -c(1:3, which(names(data) %in% c("Macrophages M1", "P-value","Correlation", "RMSE", "Absolute score (sig.score)")))]

Y <- data$'Macrophages M1'

# Fit the Partial Least Squares Regression model

pls.model <- plsr(Y ~ ., data = cbind(Y, X), ncomp = 10, validation = "CV")

# Summary of the model

summary(pls.model)

# For plots and diagnostics

plot(pls.model)

# For a fitting curve, you need to plot observed vs. predicted values

# Extract the predicted values from the model

predictions <- predict(pls.model, ncomp = 10)

# Extract the observed values (assuming 'response' is your response variable)

observed <- data$'Macrophages M1'

shape\_var <- ifelse(data$Tissue == "lung", 3, 1)

# Plot the points with different shapes

plot(observed, predictions, pch = shape\_var, xlab = "Observed Values", ylab = "Predicted Values", main = "M1 Macrophages predicted vs. actual values")

# Add legend

legend("topleft", legend = c("lung", "GI"), pch = c(3, 1))

#fitting curve

abline(0, 1, col = "red")

# Assuming cross-validation has been performed on the `pls\_model`

validationplot(pls.model, val.type = "MSEP")

# Assuming `actual\_values` are your observed Y values and `pls\_model` is your fitted model.

# Replace `pls\_model` with the actual variable name of your fitted PLS model.

# Step 1: Extract predicted values from the cross-validation (assuming the model has them)

predicted\_values <- predict(pls.model, ncomp = 10, newdata = cbind(Y, X), validation = "CV")

# Step 2: Calculate PRESS

PRESS <- sum((observed - predictions)^2)

# Step 3: Calculate TSS

TSS <- sum((observed - mean(observed))^2)

# Step 4: Calculate Q^2

Q2 <- 1 - (PRESS/TSS)

# Print Q2 value

print(Q2)

#############VIP-COEFFICIENT GENERATION#####################

library(vip)

# Assuming your PLS model is named 'pls\_model'

vip\_scores <- vip(pls.model, method = "model", num\_features = ncol(pls.model$model))

Most\_important\_vip\_scores <- vip(pls.model, method = "model")

plot(vip\_scores)

print(vip\_scores)

# Assuming vip\_scores$data contains the numerical values of the VIP scores

vip\_data <- vip\_scores$data

# Now write the data frame to a CSV file

write.csv(vip\_data, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/M1-PLS\_VIP.csv", row.names = TRUE)

coefficients <- coef(pls.model, ncomp = 10)

# Write the matrix to a CSV file.

write.csv(coefficients, file = "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/M1-PLS\_Coefficients.csv",row.names = TRUE)

##Combination of VIP and Coefficient######

# Load the necessary package for writing Excel files

install.packages("writexl")

library(writexl)

# Read the CSV files

vip\_file <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/M1-PLS\_VIP.csv", stringsAsFactors = FALSE)

coef\_file <- read.csv("C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/M1-PLS\_Coefficients.csv", stringsAsFactors = FALSE)

# Merge the two data frames by the 'Gene' column

combined\_data <- merge(vip\_file, coef\_file, by = "Cell", all = TRUE)

# Check the first few rows of the combined data frame

head(combined\_data)

# Write the combined data frame to a new Excel file

write\_xlsx(combined\_data, "C:/Users/Hu Huiqian/Box/Huiqian Hu/STAR Protocols/Cell Rep data/VIP-Coefficients.xlsx")

#Plot VIP-Coefficient##############################################################################################################################

library(ggrepel)

library(ggplot2)

# Create a new column to determine if the point is above the threshold

combined\_data$above\_threshold <- combined\_data$Importance > 0.04

ggplot(combined\_data) +

# Points below the threshold, no labels, default color

geom\_point(aes(x = Y.10.comps, y = Importance), color = "grey", data = subset(combined\_data, above\_threshold == FALSE)) +

# Points above the threshold with labels, red color

geom\_point(aes(x = Y.10.comps, y = Importance), color = "red", data = subset(combined\_data, above\_threshold == TRUE)) +

geom\_text\_repel(aes(x = Y.10.comps, y = Importance, label = ifelse(above\_threshold, as.character(Cell), "")),

box.padding = 0.35, point.padding = 0.5,

data = subset(combined\_data, above\_threshold == TRUE)) +

theme\_minimal() +

labs(x = "Coefficients", y = "VIP Scores", title = "VIP Scores vs. Coefficients") +

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

geom\_hline(yintercept = 0.04, linetype = "dashed", color = "blue") +

geom\_vline(xintercept = 0, linetype = "dashed", color = "black")+ # This is the y=0.04 line

theme(

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

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

panel.background = element\_blank(),

panel.border = element\_rect(colour = "black", fill=NA, size=1)

) # Ensure this parenthesis is closed before the library call