

```

# =====
# R Script: Integrated DEG & GO Enrichment Analysis (GDS2778)
# =====

# --- 1. SETUP ENVIRONMENT ---
# Set the working directory to save all plots and data automatically
# Note: Ensure the directory exists or create it before running
setwd("D:/DOWNLOADS")
print(paste("Working directory set to:", getwd()))

# Install required packages if not already installed
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")

BiocManager::install(c("GEOquery", "limma", "clusterProfiler", "org.Hs.eg.db",
  "enrichplot", "pheatmap", "RColorBrewer", "rmarkdown"))

# Load libraries
library(GEOquery)
library(limma)
library(clusterProfiler)
library(org.Hs.eg.db)
library(enrichplot)
library(pheatmap)
library(RColorBrewer)

# --- 2. DATA RETRIEVAL (GDS2778) ---
print("--- Downloading Data for GDS2778 ---")
gds <- getGEO("GDS2778") # Download GDS2778 data from GEO
eset <- GDS2eSet(gds)    # Convert to ExpressionSet

# Inspect data
print("--- Phenotype Data (First 2 Columns) ---")
print(pData(eset)[, 1:2])
print("--- Expression Matrix Dimensions ---")
print(dim(exprs(eset)))
print("--- First 4 Rows of Expression Data ---")
print(exprs(eset)[1:4, ])

# --- 3. DIFFERENTIAL GENE EXPRESSION (DEG) ANALYSIS ---

# Define experimental design
targets <- pData(eset)
# Create design matrix based on 'agent' column (Treatment vs Control)
design <- model.matrix(~ -1 + factor(as.character(targets$agent)))
colnames(design) <- c("Treatment", "Control")

# Fit linear model
fit <- lmFit(eset, design)

# Define contrast (Control - Treatment to check effect of treatment relative to control)
contrast.matrix <- makeContrasts(Control - Treatment, levels = design)
fit2 <- contrasts.fit(fit, contrast.matrix)
fit2 <- eBayes(fit2)

# Extract top differentially expressed genes
deg_df <- topTable(fit2, coef = 1, adjust = "fdr", sort.by = "B", number = Inf)

# Remove rows with NA adjusted p-values
deg_df <- deg_df[!is.na(deg_df$adj.P.Val), ]

# Inspect results
print("--- Top DEGs (Columns 22-27) ---")

```

```

print(head(deg_df)[, 22:27])

# Save full DEG results to CSV
write.csv(deg_df, file = "GDS2778_DEG_results.csv")
print("✓ Saved: GDS2778_DEG_results.csv")

# --- 4. DEG FILTERING ---

# Count total genes
nr <- nrow(deg_df)
print(paste("Total genes:", nr))

# Filter by adjusted p-value <= 0.01
pf <- nrow(deg_df[deg_df$adj.P.Val <= 0.01, ])
print(paste("Genes with adj.P.Val <= 0.01:", pf))

# Filter by logFC >= 1 or <= -1
fcf <- nrow(deg_df[deg_df$logFC >= 1 | deg_df$logFC <= -1, ])
print(paste("Genes with |logFC| >= 1:", fcf))

# Combined filter (Significant & High Fold Change)
pf_log <- deg_df$adj.P.Val <= 0.01
fcf_log <- deg_df$logFC >= 1 | deg_df$logFC <= -1
comb_filter <- pf_log & fcf_log
combf <- nrow(deg_df[comb_filter, ])
print(paste("Genes passing both filters:", combf))

# Save filtered DEGs
filtered_degs <- deg_df[comb_filter, ]
write.csv(filtered_degs, file = "GDS2778_Filtered_DEGs.csv")
print("✓ Saved: GDS2778_Filtered_DEGs.csv")

# --- 5. ENRICHMENT ANALYSIS: MOLECULAR FUNCTION (MF) ---

# Prepare gene IDs for enrichment (Filter: adj.P.Val <= 0.05)
cutoff <- 0.05
ids <- deg_df[deg_df$adj.P.Val <= cutoff, "Gene.ID"]
ids <- ids[!grepl("/", ids)] # Remove ambiguous mappings
ids <- ids[nchar(ids) != 0] # Remove empty slots

# Perform GO Enrichment for Molecular Function
ego_mf <- enrichGO(gene = ids,
                      OrgDb = org.Hs.eg.db,
                      ont = "MF",
                      pAdjustMethod = "BH",
                      pvalueCutoff = 0.1,
                      readable = TRUE)

print("--- Dimensions of MF Enrichment Results ---")
print(dim(ego_mf))
print(head(ego_mf))

# Visualization: MF Barplot
png(file = "GDS2778_GO_Enrichment_MF_Barplot.png", width = 800, height = 600)
print(barplot(ego_mf, showCategory = 10, title = "GDS2778 - GO Enrichment (MF)"))
dev.off()
print("✓ Saved: GDS2778_GO_Enrichment_MF_Barplot.png")

# Visualization: MF GO Plot (DAG)
# FIX: Calculate term similarity matrix first to prevent empty plot
if (nrow(ego_mf) > 0) {
  ego_mf <- pairwise_termsim(ego_mf)
}

```

```

png(file = "GDS2778_GO_Enrichment_MF_DAG.png", width = 1000, height = 800)
print(goplot(ego_mf)) # Use print() to ensure rendering
dev.off()
print("✓ Saved: GDS2778_GO_Enrichment_MF_DAG.png")
} else {
  print("⚠ No significant MF terms found to plot.")
}

# --- 6. ENRICHMENT ANALYSIS: BIOLOGICAL PROCESS (BP) ---

# Perform GO Enrichment for Biological Process
ego_bp <- enrichGO(gene = ids,
                     OrgDb = org.Hs.eg.db,
                     ont = "BP",
                     pAdjustMethod = "BH",
                     pvalueCutoff = 0.1,
                     readable = TRUE)

print("--- Dimensions of BP Enrichment Results ---")
print(dim(ego_bp))
print(head(ego_bp))

# Visualization: BP Barplot
png(file = "GDS2778_GO_Enrichment_BP_Barplot.png", width = 800, height = 600)
print(barplot(ego_bp, showCategory = 10, title = "GDS2778 - GO Enrichment
(BP)"))
dev.off()
print("✓ Saved: GDS2778_GO_Enrichment_BP_Barplot.png")

# --- 7. CLUSTERING HEATMAP ---

# Subset expression matrix for significant genes (adj.P.Val <= 0.05)
cutoff <- 0.05
affy_ids <- row.names(deg_df[deg_df$adj.P.Val <= cutoff, ])
# Ensure IDs exist in the expression set
affy_ids <- affy_ids[affy_ids %in% rownames(exprs(eset))]

if(length(affy_ids) > 0) {
  deg_ma <- exprs(eset)[affy_ids, ]

  # Generate Heatmap
  png(file = "GDS2778_DEG_Heatmap.png", width = 800, height = 800)
  pheatmap(deg_ma,
            scale = "row",
            color = brewer.pal(9, "Blues"),
            main = "GDS2778 - Significant DEGs Heatmap")
  dev.off()
  print("✓ Saved: GDS2778_DEG_Heatmap.png")
} else {
  print("⚠ No significant genes found for heatmap plotting.")
}

print("--- Script Completed Successfully ---")

```