"Analysis of TCGA-BRCA dataset"

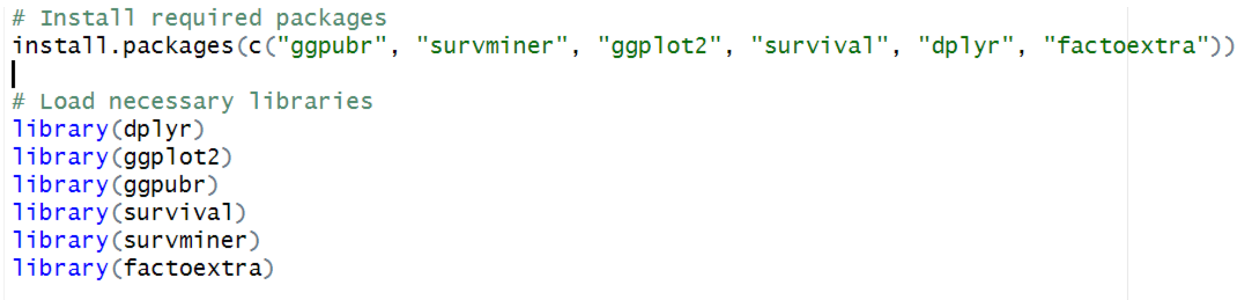
Introduction:

This report details the analysis of the TCGA-BRCA dataset obtained from cBioPortal. The analyses include data preprocessing, gene expression data, performing feature selection, clustering, and survival analysis to identify the impact of gene expression on survival patterns.

1. Data Loading and Preparation

a) Loading Required Libraries:

Explanation: Necessary libraries for data manipulation, visualization, survival analysis, and clustering are installed and loaded.

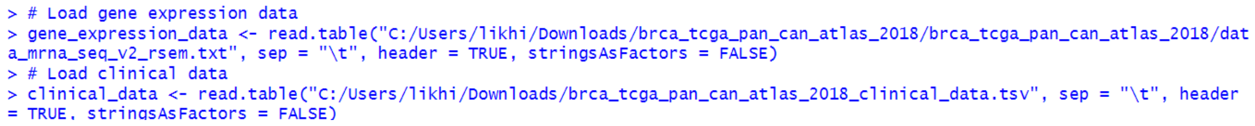


```r
# Install required packages
install.packages(c("ggpubr", "survminer", "ggplot2", "survival", "dplyr", "factoextra"))

# Load necessary libraries
library(dplyr)
library(ggplot2)
library(ggpubr)
library(survival)
library(survminer)
library(factoextra)
```

b) Loading gene expression and Clinical Data:

Read the Gene expression and clinical data from the specified file paths into R data frames.

Code:



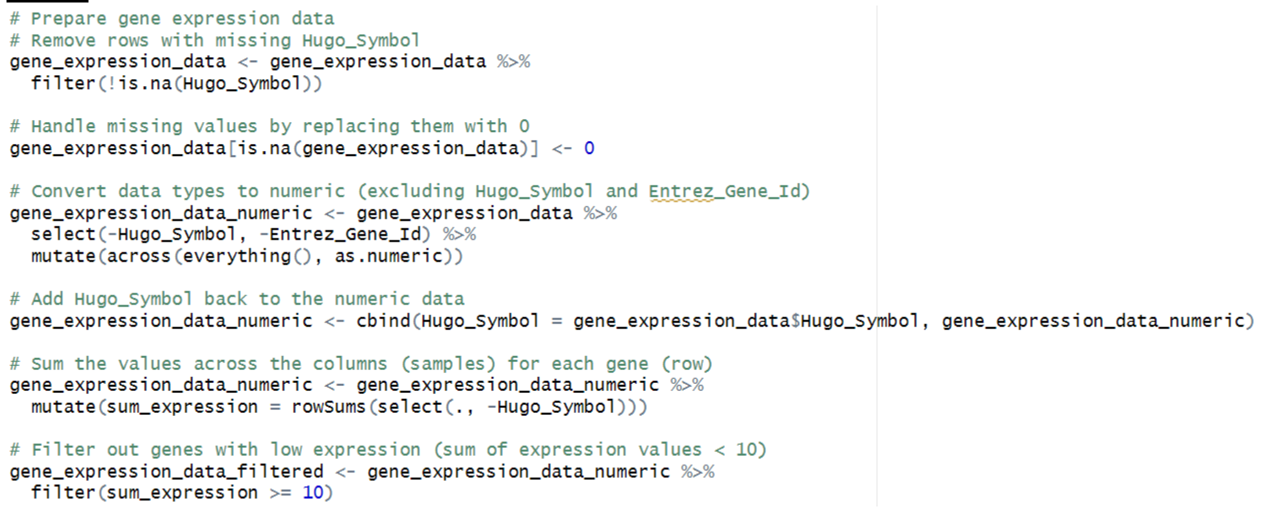
```
> # Load gene expression data
> gene_expression_data <- read.table("C:/Users/likhi/Downloads/brca_tcga_pan_can_atlas_2018/brca_tcga_pan_can_atlas_2018/dat
a_mrna_seq_v2_rsem.txt", sep = "\t", header = TRUE, stringsAsFactors = FALSE)
> # Load clinical data
> clinical_data <- read.table("C:/Users/likhi/Downloads/brca_tcga_pan_can_atlas_2018_clinical_data.tsv", sep = "\t", header
= TRUE, stringsAsFactors = FALSE)
```

2. Gene expression data preparation/ Reading and preprocessing of the data

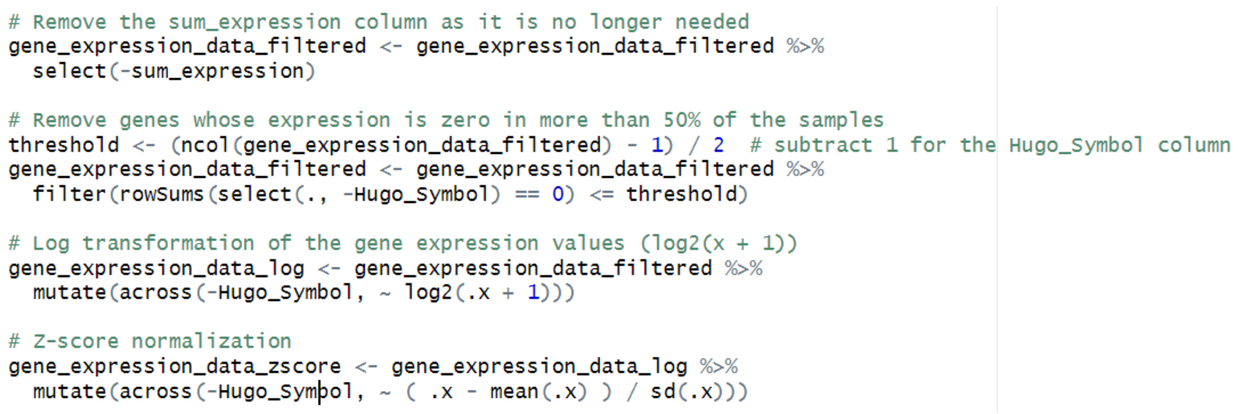
The preprocessing of gene expression data is a crucial step for ensuring the data's quality and reliability. This process involves several key steps.

i. **Filter and convert Data Types:** Remove irrelevant data and ensure variables are in appropriate formats.
ii. **Filter low-expression genes**: Remove genes with very few or no counts to reduce noise.
iii. **Normalize data:** Normalize gene expression levels to make them comparable.
iv. **Transpose data for analysis**: Adjust data orientation for analysis.
v. **Prepare clinical data**: Ensure clinical data is organized and aligned with gene expression data.
vi. **Merge data:** Integrate the gene expression and clinical data.

Code:

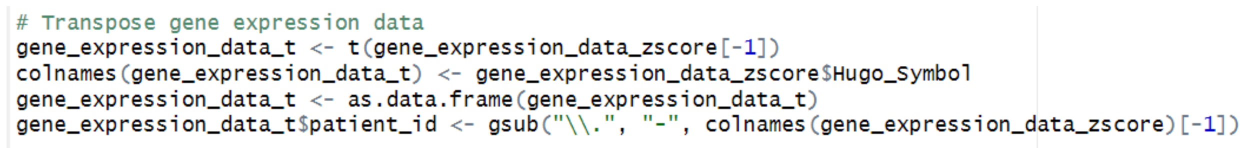


```
# Prepare gene expression data
# Remove rows with missing Hugo_Symbol
gene_expression_data <- gene_expression_data %>%
  filter(!is.na(Hugo_Symbol))

# Handle missing values by replacing them with 0
gene_expression_data[is.na(gene_expression_data)] <- 0

# Convert data types to numeric (excluding Hugo_Symbol and Entrez_Gene_Id)
gene_expression_data_numeric <- gene_expression_data %>%
  select(-Hugo_Symbol, -Entrez_Gene_Id) %>%
  mutate(across(everything(), as.numeric))

# Add Hugo_Symbol back to the numeric data
gene_expression_data_numeric <- cbind(Hugo_Symbol = gene_expression_data$Hugo_Symbol, gene_expression_data_numeric)

# Sum the values across the columns (samples) for each gene (row)
gene_expression_data_numeric <- gene_expression_data_numeric %>%
  mutate(sum_expression = rowSums(select(., -Hugo_Symbol)))

# Filter out genes with low expression (sum of expression values < 10)
gene_expression_data_filtered <- gene_expression_data_numeric %>%
  filter(sum_expression >= 10)
```



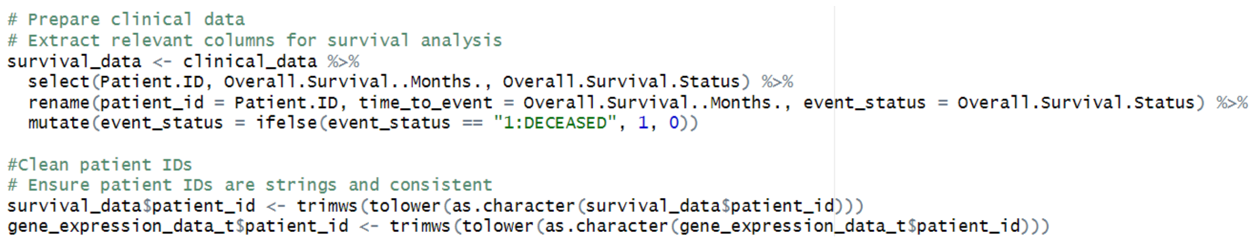
```
# Remove the sum_expression column as it is no longer needed
gene_expression_data_filtered <- gene_expression_data_filtered %>%
  select(-sum_expression)

# Remove genes whose expression is zero in more than 50% of the samples
threshold <- (ncol(gene_expression_data_filtered) - 1) / 2  # subtract 1 for the Hugo_Symbol column
gene_expression_data_filtered <- gene_expression_data_filtered %>%
  filter(rowSums(select(., -Hugo_Symbol) == 0) <= threshold)

# Log transformation of the gene expression values (log2(x + 1))
gene_expression_data_log <- gene_expression_data_filtered %>%
  mutate(across(-Hugo_Symbol, ~ log2(.x + 1)))

# Z-score normalization
gene_expression_data_zscore <- gene_expression_data_log %>%
  mutate(across(-Hugo_Symbol, ~ ( .x - mean(.x) ) / sd(.x)))
```

Transpose data for analysis



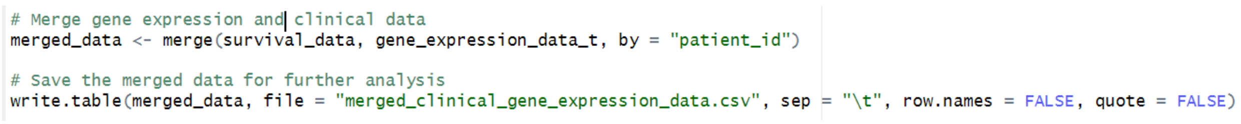
```
# Transpose gene expression data
gene_expression_data_t <- t(gene_expression_data_zscore[-1])
colnames(gene_expression_data_t) <- gene_expression_data_zscore$Hugo_Symbol
gene_expression_data_t <- as.data.frame(gene_expression_data_t)
gene_expression_data_t$patient_id <- gsub("\\.", "-", colnames(gene_expression_data_zscore)[-1])
```

Clinical data Preparation

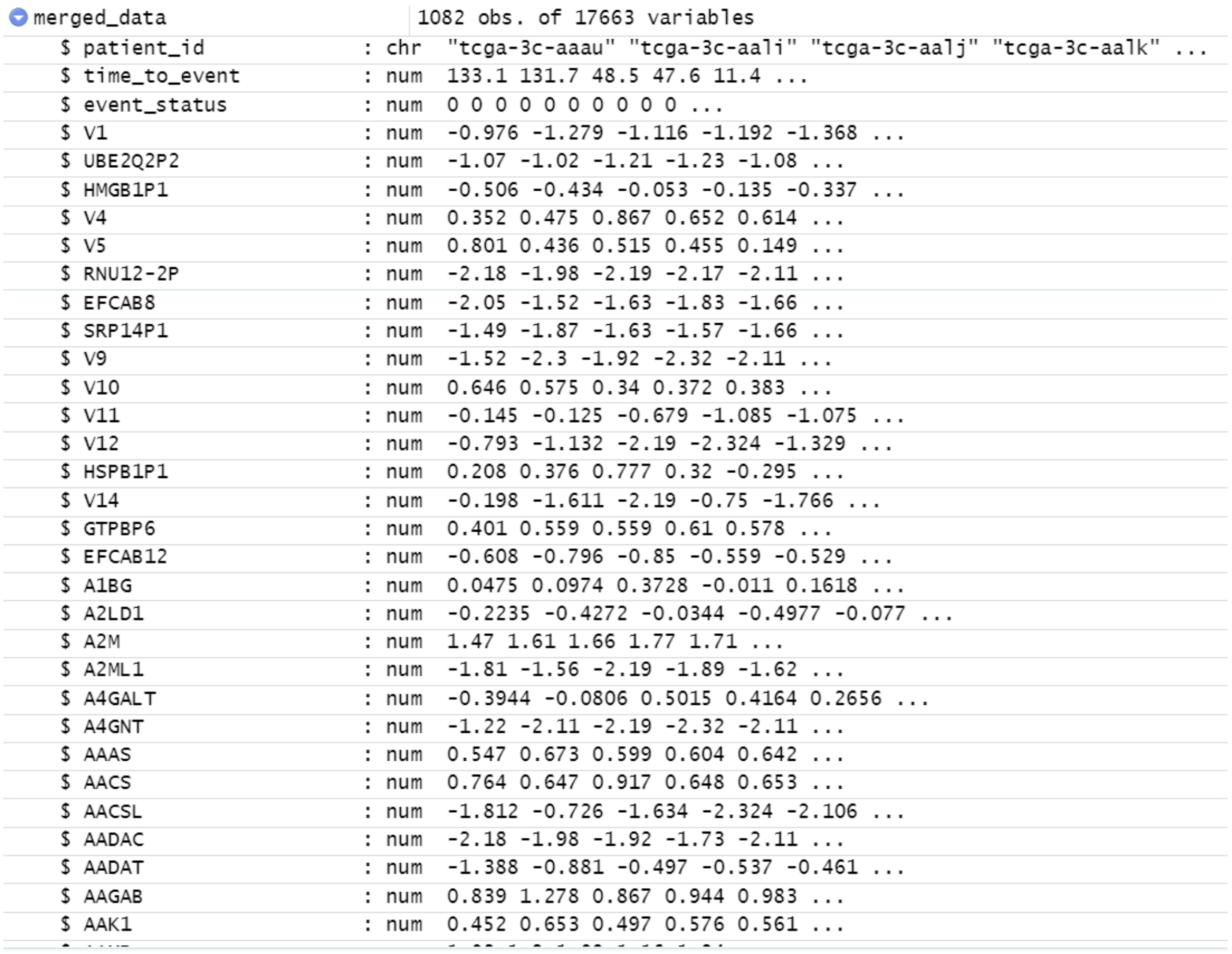


```
# Prepare clinical data
# Extract relevant columns for survival analysis
survival_data <- clinical_data %>%
  select(Patient.ID, Overall.Survival..Months., Overall.Survival.Status) %>%
  rename(patient_id = Patient.ID, time_to_event = Overall.Survival..Months., event_status = Overall.Survival.Status) %>%
  mutate(event_status = ifelse(event_status == "1:DECEASED", 1, 0))

#Clean patient IDs
# Ensure patient IDs are strings and consistent
survival_data$patient_id <- trimws(tolower(as.character(survival_data$patient_id)))
gene_expression_data_t$patient_id <- trimws(tolower(as.character(gene_expression_data_t$patient_id)))
```

Merge data



```
# Merge gene expression and clinical data
merged_data <- merge(survival_data, gene_expression_data_t, by = "patient_id")

# Save the merged data for further analysis
write.table(merged_data, file = "merged_clinical_gene_expression_data.csv", sep = "\t", row.names = FALSE, quote = FALSE)
```



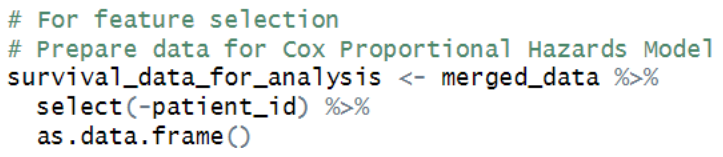
```
merged_data                      1082 obs. of 17663 variables
    $ patient_id     : chr  "tcga-3c-aaau" "tcga-3c-aali" "tcga-3c-aalj" "tcga-3c-aalk" ...
    $ time_to_event  : num  133.1 131.7 48.5 47.6 11.4 ...
    $ event_status   : num  0 0 0 0 0 0 0 0 0 0 ...
    $ V1             : num  -0.976 -1.279 -1.116 -1.192 -1.368 ...
    $ UBE2Q2P2       : num  -1.07 -1.02 -1.21 -1.23 -1.08 ...
    $ HMGB1P1        : num  -0.506 -0.434 -0.053 -0.135 -0.337 ...
    $ V4             : num  0.352 0.475 0.867 0.652 0.614 ...
    $ V5             : num  0.801 0.436 0.515 0.455 0.149 ...
    $ RNU12-2P       : num  -2.18 -1.98 -2.19 -2.17 -2.11 ...
    $ EFCAB8         : num  -2.05 -1.52 -1.63 -1.83 -1.66 ...
    $ SRP14P1        : num  -1.49 -1.87 -1.63 -1.57 -1.66 ...
    $ V9             : num  -1.52 -2.3 -1.92 -2.32 -2.11 ...
    $ V10            : num  0.646 0.575 0.34 0.372 0.383 ...
    $ V11            : num  -0.145 -0.125 -0.679 -1.085 -1.075 ...
    $ V12            : num  -0.793 -1.132 -2.19 -2.324 -1.329 ...
    $ HSPB1P1        : num  0.208 0.376 0.777 0.32 -0.295 ...
    $ V14            : num  -0.198 -1.611 -2.19 -0.75 -1.766 ...
    $ GTPBP6         : num  0.401 0.559 0.559 0.61 0.578 ...
    $ EFCAB12        : num  -0.608 -0.796 -0.85 -0.559 -0.529 ...
    $ A1BG           : num  0.0475 0.0974 0.3728 -0.011 0.1618 ...
    $ A2LD1          : num  -0.2235 -0.4272 -0.0344 -0.4977 -0.077 ...
    $ A2M            : num  1.47 1.61 1.66 1.77 1.71 ...
    $ A2ML1          : num  -1.81 -1.56 -2.19 -1.89 -1.62 ...
    $ A4GALT         : num  -0.3944 -0.0806 0.5015 0.4164 0.2656 ...
    $ A4GNT          : num  -1.22 -2.11 -2.19 -2.32 -2.11 ...
    $ AAAS           : num  0.547 0.673 0.599 0.604 0.642 ...
    $ AACS           : num  0.764 0.647 0.917 0.648 0.653 ...
    $ AACSL          : num  -1.812 -0.726 -1.634 -2.324 -2.106 ...
    $ AADAC          : num  -2.18 -1.98 -1.92 -1.73 -2.11 ...
    $ AADAT          : num  -1.388 -0.881 -0.497 -0.537 -0.461 ...
    $ AAGAB          : num  0.839 1.278 0.867 0.944 0.983 ...
    $ AAK1           : num  0.452 0.653 0.497 0.576 0.561 ...
```

This is the result of the merged data

3. Feature selection

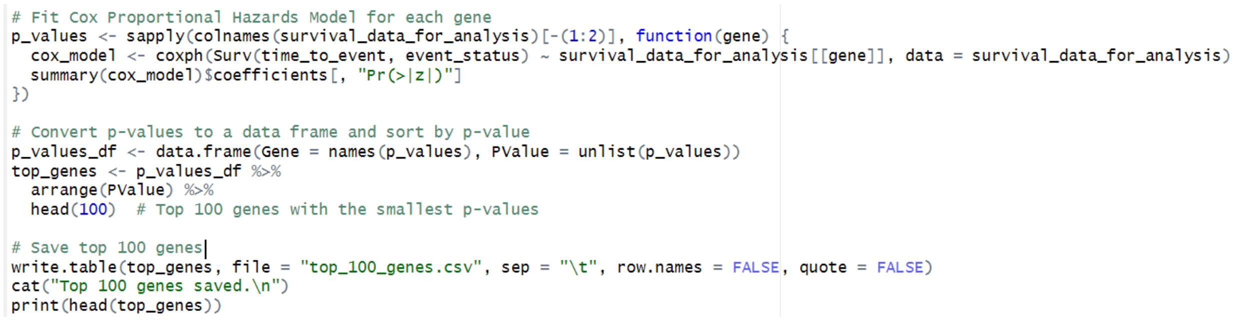
- Perform feature selection to Identify the top 100 genes associated with survival outcomes by using the Cox-Proportional Hazards model.

<u>Code:</u>



```
# For feature selection
# Prepare data for Cox Proportional Hazards Model
survival_data_for_analysis <- merged_data %>%
  select(-patient_id) %>%
  as.data.frame()
```

1-Fit Cox Proportional Hazards Model



```r
# Fit Cox Proportional Hazards Model for each gene
p_values <- sapply(colnames(survival_data_for_analysis)[-(1:2)], function(gene) {
  cox_model <- coxph(Surv(time_to_event, event_status) ~ survival_data_for_analysis[[gene]], data = survival_data_for_analysis)
  summary(cox_model)$coefficients[, "Pr(>|z|)"]
})

# Convert p-values to a data frame and sort by p-value
p_values_df <- data.frame(Gene = names(p_values), PValue = unlist(p_values))
top_genes <- p_values_df %>%
  arrange(PValue) %>%
  head(100)  # Top 100 genes with the smallest p-values

# Save top 100 genes
write.table(top_genes, file = "top_100_genes.csv", sep = "\t", row.names = FALSE, quote = FALSE)
cat("Top 100 genes saved.\n")
print(head(top_genes))
```

The top 100 genes were selected based on the Cox proportional hazards model.

4. Clustering:

Perform clustering using the K-means algorithm

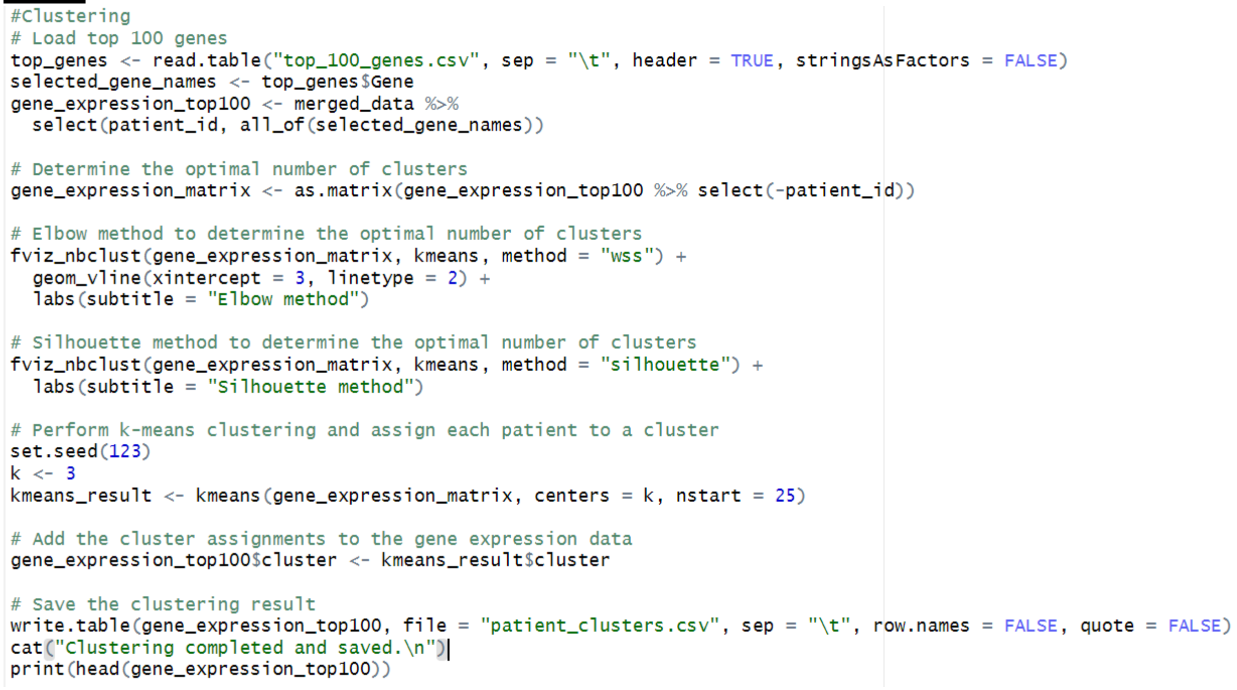
Cluster patients based on the top 100 genes to identify patterns in gene expression.

- Added steps to determine the optimal number of clusters.

Steps:

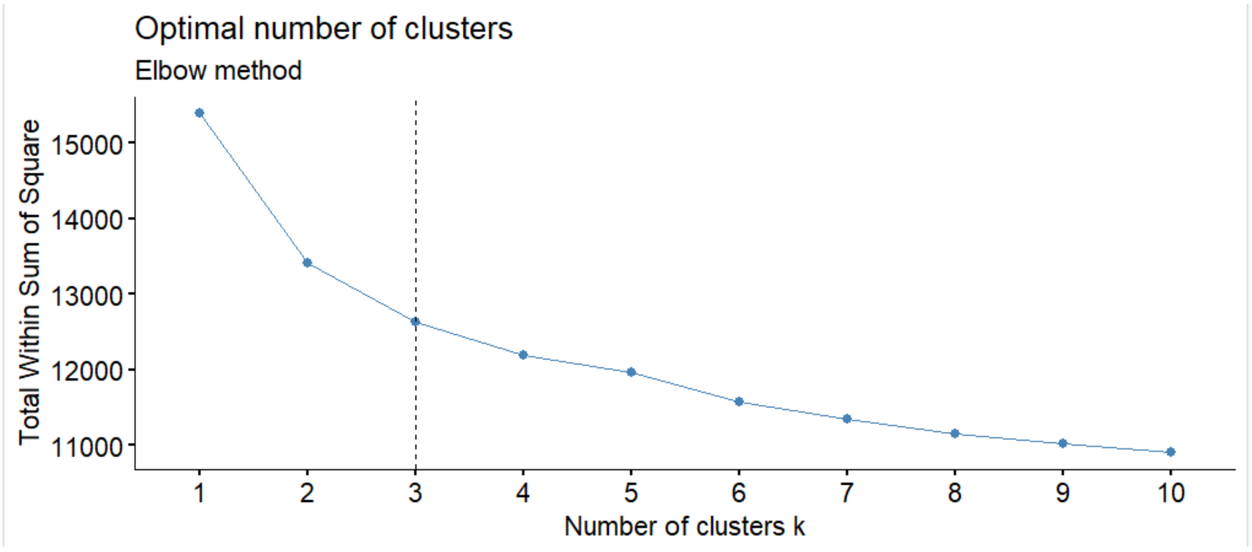
1. Load the top 100 genes
2. Determine the Optimal Number of Clusters (convert data for clustering; elbow method; silhouette method)

Code:

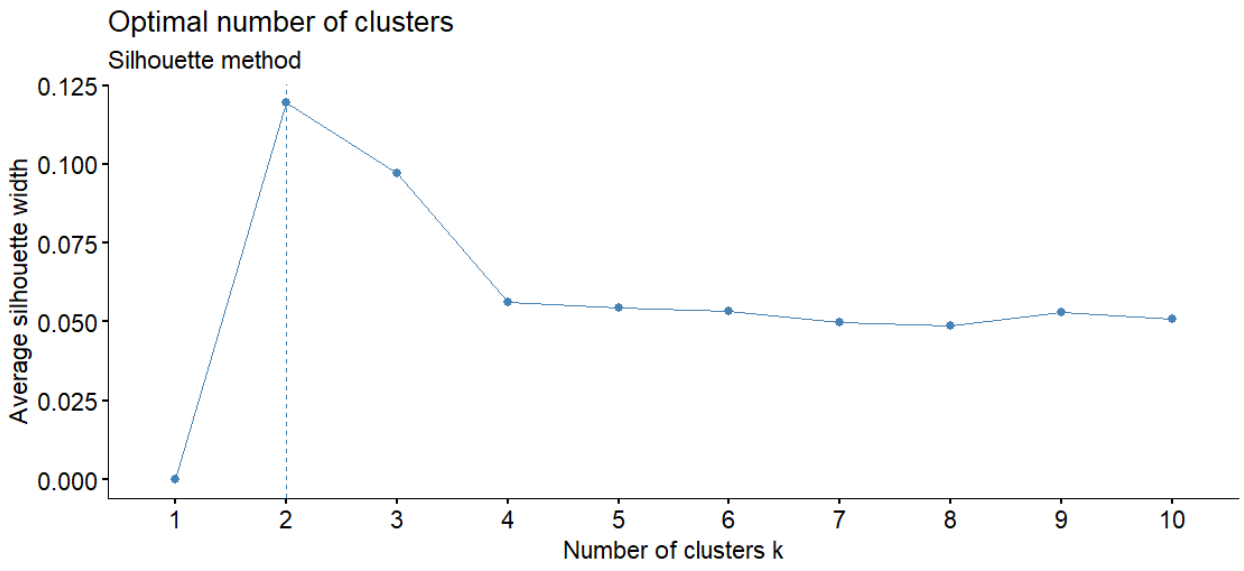


```r
#Clustering
# Load top 100 genes
top_genes <- read.table("top_100_genes.csv", sep = "\t", header = TRUE, stringsAsFactors = FALSE)
selected_gene_names <- top_genes$Gene
gene_expression_top100 <- merged_data %>%
  select(patient_id, all_of(selected_gene_names))

# Determine the optimal number of clusters
gene_expression_matrix <- as.matrix(gene_expression_top100 %>% select(-patient_id))

# Elbow method to determine the optimal number of clusters
fviz_nbclust(gene_expression_matrix, kmeans, method = "wss") +
  geom_vline(xintercept = 3, linetype = 2) +
  labs(subtitle = "Elbow method")

# Silhouette method to determine the optimal number of clusters
fviz_nbclust(gene_expression_matrix, kmeans, method = "silhouette") +
  labs(subtitle = "Silhouette method")

# Perform k-means clustering and assign each patient to a cluster
set.seed(123)
k <- 3
kmeans_result <- kmeans(gene_expression_matrix, centers = k, nstart = 25)

# Add the cluster assignments to the gene expression data
gene_expression_top100$cluster <- kmeans_result$cluster

# Save the clustering result
write.table(gene_expression_top100, file = "patient_clusters.csv", sep = "\t", row.names = FALSE, quote = FALSE)
cat("Clustering completed and saved.\n")
print(head(gene_expression_top100))
```

Result of the Elbow method:



Result of the Silhouette method:



Explanation: Patients were clustered into three groups based on the top 100 gene expressions using K-means clustering. The optimal number of clusters was determined using the elbow and silhouette methods. Clustering completed and saved.

5. Survival Analysis in clusters

Randomly pick one of the top 100 genes. Here I have selected a random gene 66 (HES5) and conducted survival analysis within clusters using the Cox Proportional Hazards Model and Kaplan-Meier survival plots to analyze survival within each cluster and assess the impact of gene expression.

Code:

```
# Randomly select one gene from the top 100 genes
set.seed(123)
selected_gene <- selected_gene_names[66] #select 66th gene from the list of top 100
cat("Selected Gene:", selected_gene, "\n")
```

a) Cox Proportional Hazards Model (CoxPH):

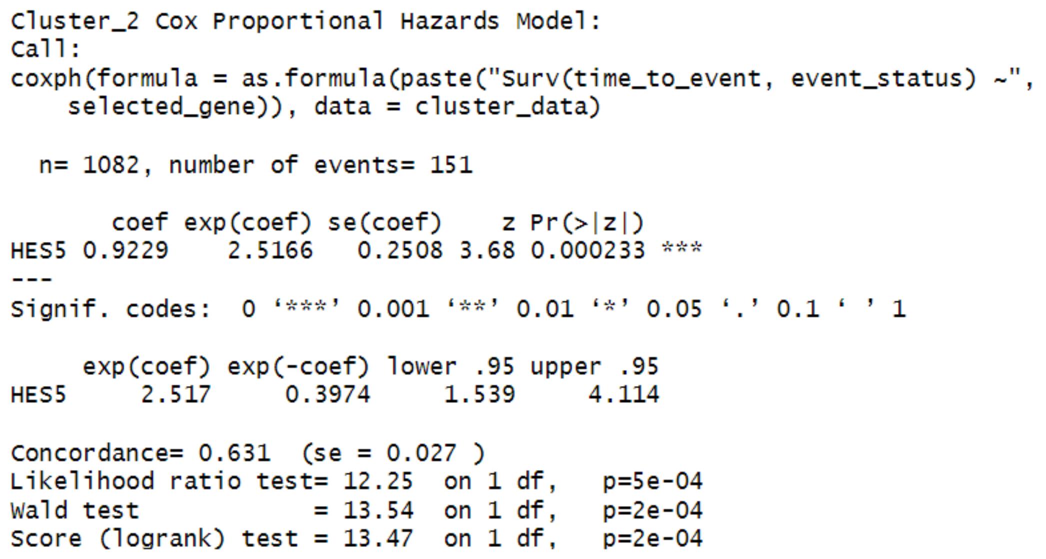
```
#Survival analysis in clusters
# Fit Cox Proportional Hazards Model within each cluster
coxph_results <- list()

for (cluster in unique(gene_expression_top100$cluster)) {
  cluster_data <- merged_data %>%
    filter(cluster == cluster)

  if (nrow(cluster_data) == 0 || sum(cluster_data$event_status) == 0) {
    cat("No events observed in Cluster", cluster, ". Skipping Cox Proportional Hazards Model.\n")
    next
  }

  #Fit the cox model
  cox_model <- coxph(as.formula(paste("Surv(time_to_event, event_status) ~", selected_gene)), data = cluster_data)
  coxph_results[[paste0("Cluster_", cluster)]] <- summary(cox_model)
}

# Print Cox Proportional Hazards Model results
for (cluster in names(coxph_results)) {
  cat(cluster, "Cox Proportional Hazards Model:\n")
  print(coxph_results[[cluster]])
}
```

Analysis of the selected gene's impact on survival within each cluster by CoxPH model:

Cluster_1

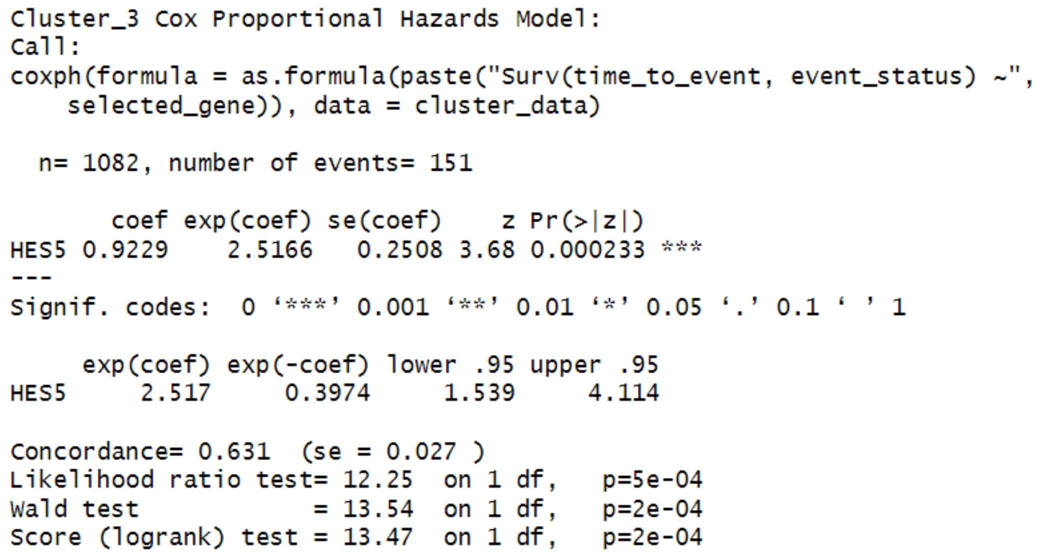
```
Cluster_1 Cox Proportional Hazards Model:
Call:
coxph(formula = as.formula(paste("Surv(time_to_event, event_status) ~",
    selected_gene)), data = cluster_data)

  n= 1082, number of events= 151

       coef exp(coef) se(coef)    z Pr(>|z|)
HES5 0.9229    2.5166   0.2508 3.68 0.000233 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     exp(coef) exp(-coef) lower .95 upper .95
HES5     2.517     0.3974     1.539     4.114

Concordance= 0.631  (se = 0.027 )
Likelihood ratio test= 12.25  on 1 df,   p=5e-04
Wald test            = 13.54  on 1 df,   p=2e-04
Score (logrank) test = 13.47  on 1 df,   p=2e-04
```

Cluster_2



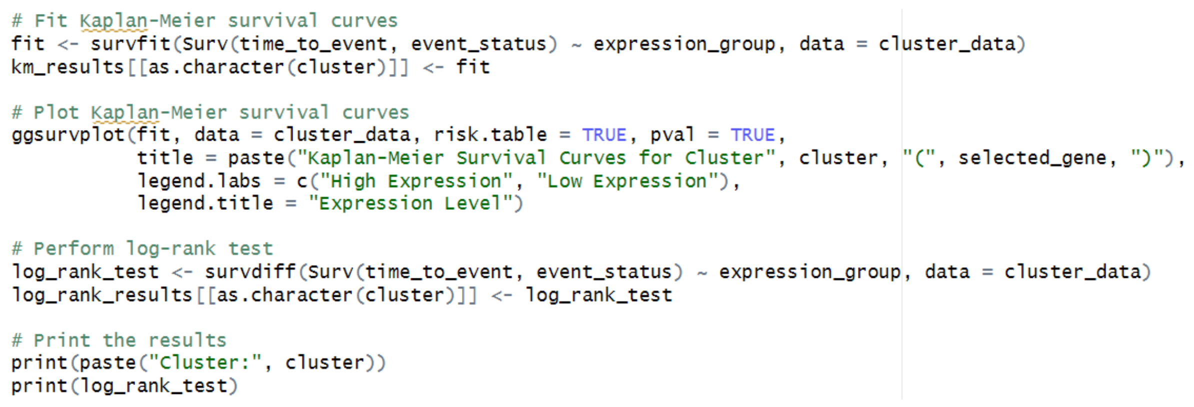
```
Cluster_2 Cox Proportional Hazards Model:
Call:
coxph(formula = as.formula(paste("Surv(time_to_event, event_status) ~",
    selected_gene)), data = cluster_data)

  n= 1082, number of events= 151

       coef exp(coef) se(coef)    z Pr(>|z|)
HES5 0.9229    2.5166   0.2508 3.68 0.000233 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     exp(coef) exp(-coef) lower .95 upper .95
HES5     2.517     0.3974     1.539     4.114

Concordance= 0.631  (se = 0.027 )
Likelihood ratio test= 12.25  on 1 df,   p=5e-04
Wald test            = 13.54  on 1 df,   p=2e-04
Score (logrank) test = 13.47  on 1 df,   p=2e-04
```

Cluster_3



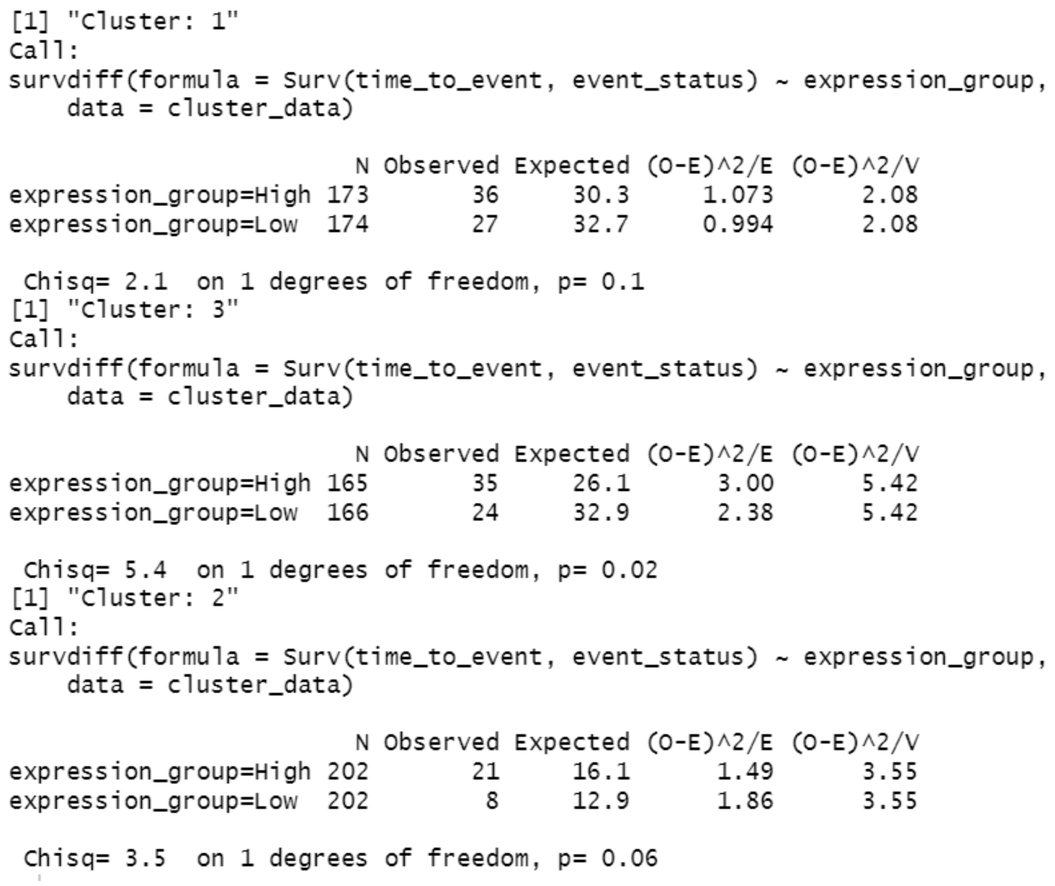
```
Cluster_3 Cox Proportional Hazards Model:
Call:
coxph(formula = as.formula(paste("Surv(time_to_event, event_status) ~",
    selected_gene)), data = cluster_data)

  n= 1082, number of events= 151

       coef exp(coef) se(coef)    z Pr(>|z|)
HES5 0.9229    2.5166   0.2508 3.68 0.000233 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     exp(coef) exp(-coef) lower .95 upper .95
HES5     2.517     0.3974     1.539     4.114

Concordance= 0.631  (se = 0.027 )
Likelihood ratio test= 12.25  on 1 df,   p=5e-04
Wald test            = 13.54  on 1 df,   p=2e-04
Score (logrank) test = 13.47  on 1 df,   p=2e-04
```

b) Kaplan-Meier Survival Analysis



```
# Fit Kaplan-Meier survival curves
fit <- survfit(Surv(time_to_event, event_status) ~ expression_group, data = cluster_data)
km_results[[as.character(cluster)]] <- fit

# Plot Kaplan-Meier survival curves
ggsurvplot(fit, data = cluster_data, risk.table = TRUE, pval = TRUE,
           title = paste("Kaplan-Meier Survival Curves for Cluster", cluster, "(", selected_gene, ")"),
           legend.labs = c("High Expression", "Low Expression"),
           legend.title = "Expression Level")

# Perform log-rank test
log_rank_test <- survdiff(Surv(time_to_event, event_status) ~ expression_group, data = cluster_data)
log_rank_results[[as.character(cluster)]] <- log_rank_test

# Print the results
print(paste("Cluster:", cluster))
print(log_rank_test)
```

Result:



```
[1] "Cluster: 1"
Call:
survdiff(formula = Surv(time_to_event, event_status) ~ expression_group,
    data = cluster_data)

                         N Observed Expected (O-E)^2/E (O-E)^2/V
expression_group=High 173       36     30.3     1.073      2.08
expression_group=Low  174       27     32.7     0.994      2.08

 Chisq= 2.1  on 1 degrees of freedom, p= 0.1
[1] "Cluster: 3"
Call:
survdiff(formula = Surv(time_to_event, event_status) ~ expression_group,
    data = cluster_data)

                         N Observed Expected (O-E)^2/E (O-E)^2/V
expression_group=High 165       35     26.1      3.00      5.42
expression_group=Low  166       24     32.9      2.38      5.42

 Chisq= 5.4  on 1 degrees of freedom, p= 0.02
[1] "Cluster: 2"
Call:
survdiff(formula = Surv(time_to_event, event_status) ~ expression_group,
    data = cluster_data)

                         N Observed Expected (O-E)^2/E (O-E)^2/V
expression_group=High 202       21     16.1      1.49      3.55
expression_group=Low  202        8     12.9      1.86      3.55

 Chisq= 3.5  on 1 degrees of freedom, p= 0.06
```

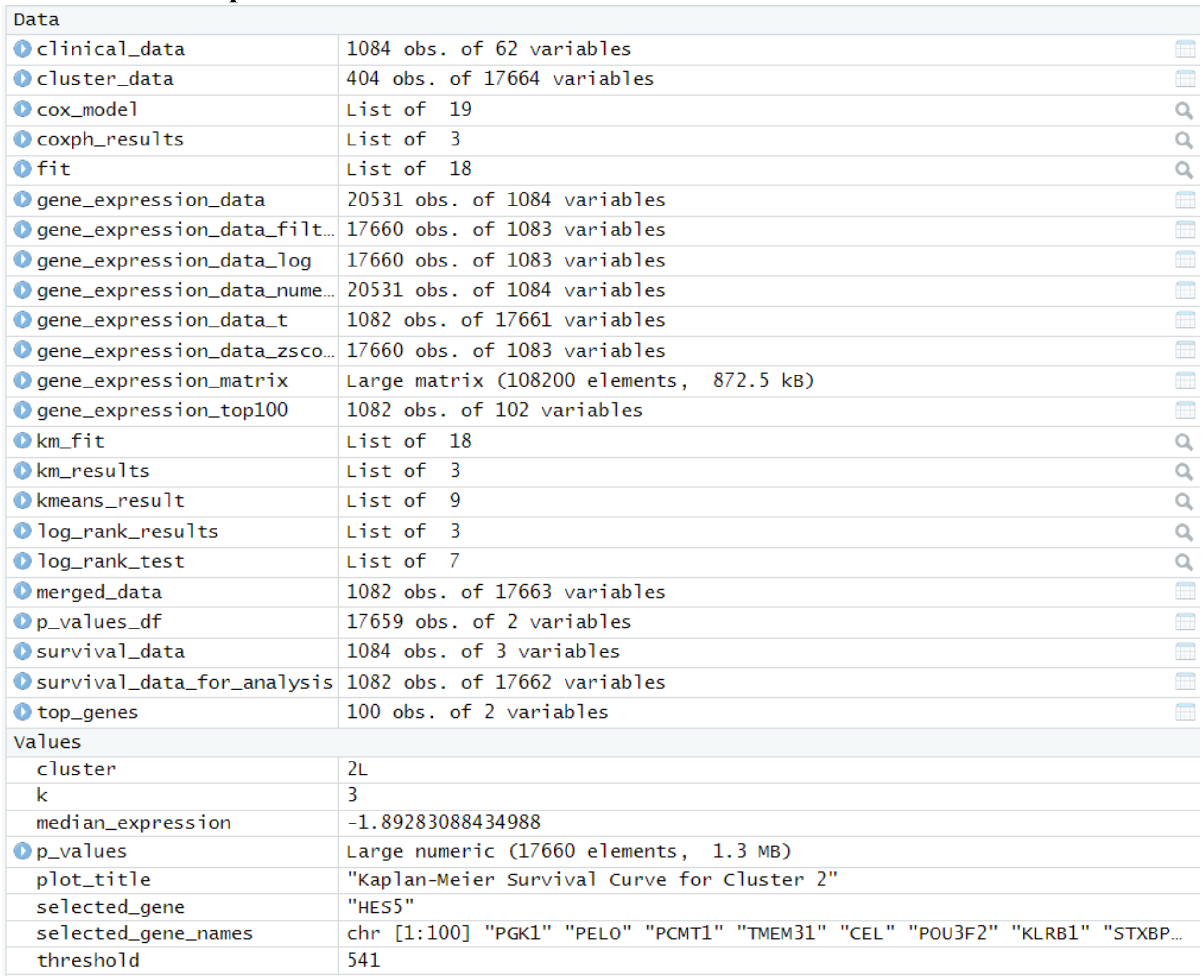
Explanation:

The results of the survival analysis using the log-rank test (survdiff function) for different clusters indicate the following:

Cluster 1: There is no statistically significant difference in survival between the high and low-expression groups, with a chi-square value of 2.1 and a p-value of 0.1. This suggests that gene expression does not significantly impact survival in this cluster.
Cluster 3: There is a statistically significant difference in survival between the high and low-expression groups, with a chi-square value of 5.4 and a p-value of 0.02. This indicates that gene expression significantly impacts survival in this cluster, with distinct survival patterns for the high and low-expression groups.
Cluster 2: There is a borderline significant difference in survival between the high and low-expression groups, with a chi-square value of 3.5 and a p-value of 0.06. This suggests a trend towards significance, indicating that gene expression might impact survival in this cluster, but the evidence is not strong enough to confirm it definitively.

Overall, these results suggest that the impact of gene expression on survival varies across clusters, being significant in Cluster 3, potentially significant in Cluster 2, and not significant in Cluster 1.

Results and Interpretation:



```
Data
clinical_data                    1084 obs. of 62 variables
cluster_data                     404 obs. of 17664 variables
cox_model                        List of  19
coxph_results                    List of  3
fit                              List of  18
gene_expression_data             20531 obs. of 1084 variables
gene_expression_data_filt…       17660 obs. of 1083 variables
gene_expression_data_log         17660 obs. of 1083 variables
gene_expression_data_nume…       20531 obs. of 1084 variables
gene_expression_data_t           1082 obs. of 17661 variables
gene_expression_data_zsco…       17660 obs. of 1083 variables
gene_expression_matrix           Large matrix (108200 elements,  872.5 kB)
gene_expression_top100           1082 obs. of 102 variables
km_fit                           List of  18
km_results                       List of  3
kmeans_result                    List of  9
log_rank_results                 List of  3
log_rank_test                    List of  7
merged_data                      1082 obs. of 17663 variables
p_values_df                      17659 obs. of 2 variables
survival_data                    1084 obs. of 3 variables
survival_data_for_analysis       1082 obs. of 17662 variables
top_genes                        100 obs. of 2 variables
Values
cluster                          2L
k                                3
median_expression                -1.89283088434988
p_values                         Large numeric (17660 elements,  1.3 MB)
plot_title                       "Kaplan-Meier Survival Curve for Cluster 2"
selected_gene                    "HES5"
selected_gene_names              chr [1:100] "PGK1" "PELO" "PCMT1" "TMEM31" "CEL" "POU3F2" "KLRB1" "STXBP…
threshold                        541
```

Conclusion:

This analysis of the TCGA-BRCA dataset revealed varying impacts of gene expression on survival outcomes across different patient clusters. Specifically, gene expression had a significant effect in Cluster 3 (p-value of 0.02), indicating distinct survival patterns based on gene expression levels. In Cluster 2, there was a borderline significant trend (p-value of 0.06), suggesting potential relevance but insufficient for definitive conclusions. In contrast, no significant difference in survival was observed in Cluster 1 (p-value of 0.1). These findings highlight that the role of gene expression in survival varies significantly between clusters, with significant effects observed in some groups but not uniformly across all clusters.