

Survival_analysis

```
# Check for relevant columns in clinical data
colnames_to_check <- c("vital_status", "days_to_last_follow_up", "days_to_death")
has_relevant_columns <- any(colnames(clinical_data_escc) %in% colnames_to_check)
relevant_columns_indices <- which(colnames(clinical_data_escc) %in% colnames_to_check)
relevant_columns <- clinical_data_escc[, relevant_columns_indices]
```

```
# Print summary of vital status
table(clinical_data_escc$vital_status)
```

```
##
## Alive   Dead
##   108    77
```

```
# Create a new variable "deceased" based on vital status
clinical_data_escc$deceased <- ifelse(clinical_data_escc$vital_status == "Alive", FALSE, TRUE)
```

```
# Create an "overall_survival" variable that considers days_to_death for deceased patients and days_to_
clinical_data_escc$overall_survival <- ifelse(clinical_data_escc$vital_status == "Alive",
                                             clinical_data_escc$days_to_last_follow_up,
                                             clinical_data_escc$days_to_death)
```

```
# Build a query to retrieve gene expression data for the entire cohort
query_escc_all <- GDCquery(
  project = "TCGA-ESCA",
  data.category = "Transcriptome Profiling",
  experimental.strategy = "RNA-Seq",
  workflow.type = "STAR - Counts",
  data.type = "Gene Expression Quantification",
  access = "open"
)
```

```
## -----
```

```
## o GDCquery: Searching in GDC database
```

```
## -----
```

```
## Genome of reference: hg38
```

```
## -----
```

```
## oo Accessing GDC. This might take a while...
```

```

## -----
## ooo Project: TCGA-ESCA
## -----
## oo Filtering results
## -----
## ooo By access
## ooo By experimental.strategy
## ooo By data.type
## ooo By workflow.type
## -----
## oo Checking data
## -----
## ooo Checking if there are duplicated cases
## ooo Checking if there are results for the query
## -----
## o Preparing output
## -----

```

```

output_escscc <- getResults(query_escscc_all)
tumor <- output_escscc$cases

```

```

# Build a query to retrieve gene expression data for 20 primary tumors and solid tissue normal samples
query_escscc <- GDCquery(
  project = "TCGA-ESCA",
  data.category = "Transcriptome Profiling",
  experimental.strategy = "RNA-Seq",
  workflow.type = "STAR - Counts",
  data.type = "Gene Expression Quantification",
  sample.type = c("Primary Tumor", "Solid Tissue Normal"),
  access = "open",
  barcode = tumor
)

```

```
## -----
## o GDCquery: Searching in GDC database
## -----
## Genome of reference: hg38
## -----
## oo Accessing GDC. This might take a while...
## -----
## ooo Project: TCGA-ESCA
## -----
## oo Filtering results
## -----
## ooo By access
## ooo By experimental.strategy
## ooo By data.type
## ooo By workflow.type
## ooo By barcode
## ooo By sample.type
## -----
## oo Checking data
## -----
## ooo Checking if there are duplicated cases
## ooo Checking if there are results for the query
## -----
## o Preparing output
## -----
```

```

# Download the data
GDCdownload(query_escc)

## Downloading data for project TCGA-ESCA

## Of the 197 files for download 197 already exist.

## All samples have been already downloaded

library(SummarizedExperiment)

# Prepare the gene expression data
tcga_escc_data <- GDCprepare(query_escc, summarizedExperiment = TRUE)

## | | 0% |

## Starting to add information to samples

## => Add clinical information to samples

## => Adding TCGA molecular information from marker papers

## => Information will have prefix 'paper_'

## esca subtype information from:doi:10.1038/nature20805

## Available assays in SummarizedExperiment :
## => unstranded
## => stranded_first
## => stranded_second
## => tpm_unstrand
## => fpkm_unstrand
## => fpkm_uq_unstrand

escc_matrix <- assay(tcga_escc_data)

# Extract gene and sample metadata from the summarizedExperiment object
gene_metadata <- as.data.frame(rowData(tcga_escc_data))
coldata <- as.data.frame(colData(tcga_escc_data))

# Merge gene expression data with gene metadata using gene_id
merged_data <- merge(escc_matrix, gene_metadata, by.x = 0, by.y = "gene_id")
test_gene <-merged_data

# Extract gene expression data for TCGA samples
sample_ids <- colnames(test_gene)

```

```

# Extract gene expression data for TCGA samples
sample_ids <- colnames(test_gene)

# Function to assign groups based on TCGA IDs
assign_group <- function(tcga_id) {
  group <- ""
  parts <- unlist(strsplit(tcga_id, "-"))

  if (length(parts) >= 4) {
    fourth_part <- parts[4]
    if (grepl("\\d{2}", fourth_part)) {
      num <- as.numeric(substr(fourth_part, 1, 2))
      if (num >= 10 & num <= 29) {
        group <- "Control"
      } else if (num >= 1 & num <= 9) {
        group <- "Cancer"
      }
    }
  }

  return(group)
}

```

```

# Assign groups to TCGA IDs
group_assignments <- sapply(sample_ids, assign_group)

# Combine group assignments with the sample data
combined_data <- as.data.frame(group_assignments)

# VST transform counts for use in survival analysis
library(DESeq2)

# Setting up countData object
dds <- DESeqDataSetFromMatrix(countData = escc_matrix,
                              colData = coldata,
                              design = ~ 1)

```

```

# Removing genes with a sum total of 10 reads across all samples
keep <- rowSums(counts(dds)) >= 10
dds <- dds[keep,]

# VST transformation
vsd <- vst(dds, blind = FALSE)
escc_matrix_vst <- assay(vsd)

# Get data for the RUVBL1 gene and add gene metadata information to it
gene_named <- escc_matrix %>%
  as.data.frame() %>%
  rownames_to_column(var = 'gene_id') %>%
  gather(key = 'case_id', value = 'counts', -gene_id) %>%
  left_join(., gene_metadata, by = "gene_id") %>%
  filter(gene_name == "ATP6V1D")

```

```

# Calculate the median value
median_value <- median(gene_named$counts)

# Assign strata based on median count
gene_named$strata <- ifelse(gene_named$counts >= median_value, "HIGH", "LOW")

# Merge clinical information with gene expression data
gene_named$case_id <- gsub('-01.*', '', gene_named$case_id)
gene_named <- merge(gene_named, clinical_data_escr, by.x = 'case_id', by.y = 'submitter_id')

# Convert days to months for overall_survival variable
gene_named$overall_survival <- gene_named$overall_survival / 30

# Fitting survival curve
fit <- survfit(Surv(overall_survival, deceased) ~ strata, data = gene_named)

# Plotting survival curves
ggsurvplot(fit,
  data = gene_named,
  pval = TRUE,
  risk.table = FALSE)

```

