Bulk RNA-seq analysis on CCLE Colon Cell Lines

Mrunali Thokadiwala

Table of contents

1 Load Libraries **2**

2 Prepare Data **6**
2.1 Reading CCLE RNA-seq Counts 6
2.1.1 Filtering and De-duplicating Protein-Coding Genes 7
2.1.2 Preparing Numeric Counts Matrix 8
2.2 Reading Metadata for CCLE RNA-seq Samples 9
2.2.1 Taking intersection of samples in metadata and counts data 9
2.2.2 Select useful metadata columns 11
2.2.3 Cleaning metadata entries 16
2.3 Save outputs 20

3 Principal Component Analysis (PCA) **20**
3.1 Variance stabilizing transformation (`vst()`) 21
3.2 PCA with `plotPCA()` 24
3.3 PCA with `prcomp()` 28
3.4 Elbow plot 30
3.5 Gene loadings analysis 32

4 Hierarchical Clustering **38**
4.1 Dividing samples into four clusters 39
4.2 Dividing samples into two clusters 43

5 Differential Expression Analysis **46**
5.1 log2FC method (using `vsd`) 46
5.2 DESeq2 method 50
5.3 Volcano Plot 55

6 Interpretation **58**
6.1 Normal Biological Functions of top 10 DEGs in each cluster: 58
6.1.1 Top 10 DEGs in Cluster 1 (High in Cluster 1): 58
6.1.2 Top 10 DEGs in Cluster 2 (High in Cluster 2): 59
6.2 Role in Large Intestine Cancer: 59
6.2.1 Cluster 1 (Stromal/Mesenchymal-like): 59
6.2.2 Cluster 2 (Epithelial/Differentiated-like): 59
6.3 Biological Difference Between Clusters: 60

1 Load Libraries

```r
library(tidyr)
library(dplyr)
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(biomaRt)
library(ggplot2)
library(patchwork)
library(scales)
library(ComplexHeatmap)
```

```
Loading required package: grid

========================================
ComplexHeatmap version 2.20.0
Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/
Github page: https://github.com/jokergoo/ComplexHeatmap
```

```
Documentation: http://jokergoo.github.io/ComplexHeatmap-reference

If you use it in published research, please cite either one:
- Gu, Z. Complex Heatmap Visualization. iMeta 2022.
- Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional
    genomic data. Bioinformatics 2016.


The new InteractiveComplexHeatmap package can directly export static
complex heatmaps into an interactive Shiny app with zero effort. Have a try!

This message can be suppressed by:
  suppressPackageStartupMessages(library(ComplexHeatmap))
========================================
```

```r
library(ggrepel)
library(EnhancedVolcano)
library(DESeq2)
```

```
Loading required package: S4Vectors

Loading required package: stats4

Loading required package: BiocGenerics


Attaching package: 'BiocGenerics'

The following objects are masked from 'package:dplyr':

    combine, intersect, setdiff, union

The following objects are masked from 'package:stats':

    IQR, mad, sd, var, xtabs

The following objects are masked from 'package:base':

    anyDuplicated, aperm, append, as.data.frame, basename, cbind,
    colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
    get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
    match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
    Position, rank, rbind, Reduce, rownames, sapply, setdiff, table,
    tapply, union, unique, unsplit, which.max, which.min


Attaching package: 'S4Vectors'

The following objects are masked from 'package:dplyr':

    first, rename

The following object is masked from 'package:tidyr':

    expand

The following object is masked from 'package:utils':

    findMatches

The following objects are masked from 'package:base':

    expand.grid, I, unname

Loading required package: IRanges


Attaching package: 'IRanges'

The following objects are masked from 'package:dplyr':

    collapse, desc, slice

Loading required package: GenomicRanges

Loading required package: GenomeInfoDb

Loading required package: SummarizedExperiment
Loading required package: MatrixGenerics

Loading required package: matrixStats


Attaching package: 'matrixStats'


The following object is masked from 'package:dplyr':

    count



Attaching package: 'MatrixGenerics'


The following objects are masked from 'package:matrixStats':

    colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
    colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
    colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
    colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
    colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
    colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
    colWeightedMeans, colWeightedMedians, colWeightedSds,
    colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
    rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
    rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
    rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
    rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
    rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
    rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
    rowWeightedSds, rowWeightedVars


Loading required package: Biobase

Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
Attaching package: 'Biobase'


The following object is masked from 'package:MatrixGenerics':

    rowMedians


The following objects are masked from 'package:matrixStats':

    anyMissing, rowMedians
```

2 Prepare Data

2.1 Reading CCLE RNA-seq Counts

Counts data -> Genes x Samples

```
#Path to gzipped Gene Cluster Text (GCT) file, tab-separated
ccle_data <-
"/Users/emrunali/bulkRNAseq/CCLE_RNAseq_genes_counts_20180929.gct.gz"

#Reading CCLE counts table from .gct.gz file
ccle_counts <- read.csv(gzfile(ccle_data), skip = 2, sep = '\t', header =
TRUE)
```

Notes:

- **Why `skip = 2`?** GCT v1.2+ starts with a version line and a dimensions line—neither are part of the table.
- `sep = '\t'` because GCT is tab-delimited
- `header = TRUE` means the first non-skipped row has column names

```
# Checking dimensions of ccle_counts dataframe
dim(ccle_counts)
```

```
[1] 56202  1021
```

2.1.1 Filtering and De-duplicating Protein-Coding Genes

There are **56202 genes** (rows) and **1021 samples** (columns) in `ccle_counts` matrix.

1. There are so many genes because:

   A) CCLE gene counts file includes **all annotated genes**, not just protein-coding ones. That means it includes protein-coding + lncRNA + pseudogenes, etc.

   For DE analysis here, we will **filter to include only protein-coding genes**.

```r
# Connect to Ensembl's human gene dataset
ensembl <- useMart("ensembl", dataset = "hsapiens_gene_ensembl")

# Query the selected dataset for protein-coding genes retrieving their
Ensembl ID and gene symbol
coding_genes_hgnc <- getBM(attributes = c('ensembl_gene_id',
'hgnc_symbol', 'gene_biotype'),
                           filters = 'biotype',
                           values = 'protein_coding',
                           mart = ensembl)

# Filter counts matrix for protein-coding genes
coding_gene_ids <- coding_genes_hgnc$hgnc_symbol
ccle_counts <- ccle_counts[ccle_counts$Description %in% coding_gene_ids,
]
ccle_counts <- data.frame(ccle_counts)
print(dim(ccle_counts))
```

```
[1] 17726  1021
```

   So now, we only have **17726 protein-coding genes** in our counts matrix.

   B) There could be **duplicated entries for a same gene/gene symbol**. The first two columns in `ccle_counts` are:

   - *Name* = Ensembl gene ID (often unique)
   - *Description* = gene symbol (can repeat across multiple Ensembl IDs)

   If this is the case, we will **de-duplicate the dataframe to include only unique gene symbols** or they would interfere with our DE analysis.

```r
#Checking for duplicated gene symbols
print(sum(duplicated(ccle_counts$Description)))
```

```
[1] 33
```

Of our protein-coding genes, 33 appear to be duplicated so we will combine them.

```
# Extract duplicated rows/gene symbols present more than once
ccle_counts_dupes <- ccle_counts[ccle_counts$Description %in%
ccle_counts$Description[duplicated(ccle_counts$Description)], ]

#Sum numeric columns for the extracted dupes
ccle_counts_dupes_summed <- ccle_counts_dupes %>% group_by(Description)
%>% summarize(across(where(is.numeric), sum))

# Extract non-duplicated rows/gene symbols
ccle_counts_no_duplicates <-
ccle_counts[!duplicated(ccle_counts$Description) &
!duplicated(ccle_counts$Description, fromLast = TRUE), ]

# Merge the two dataframes - one with summed values for dupes and other
consisting of non-dupes from the original counts matrix
ccle_counts <- bind_rows(ccle_counts_no_duplicates,
ccle_counts_dupes_summed)
print(dim(ccle_counts))
```

```
[1] 17693  1021
```

As expected, this de-duplication resulted in **17693** (17726 - 33) **unique protein-coding genes**.

2. There are also so many samples/cell lines in this dataset, so we should check for duplicate samples as well.

```
#Checking for duplicated samples (columns)
print(sum(duplicated(colnames(ccle_counts))))
```

```
[1] 0
```

There are no duplicated samples in our dataset.

2.1.2 Preparing Numeric Counts Matrix

DESeq2 package expects a numeric matrix for counts and require gene IDs separately so it is a good practice to store gene IDs as row names to let the counts matrix remain strictly numeric. In our matrix, we have two non-numeric columns- *Name* (Ensembl ID) and *Description* (gene symbol). We will set *Description* as row names and remove the *Name* column.

```r
# Set Description as row names
rownames(ccle_counts) <- ccle_counts$Description

# Preserve Name (Ensembl IDs) for later annotation just in case
gene_map <- ccle_counts[, c("Name", "Description")]

#Drop the Name and Description columns
ccle_counts <- ccle_counts[, !(names(ccle_counts) %in% c("Name",
"Description"))]

print(dim(ccle_counts))
```

```
[1] 17693  1019
```

As expected, our `ccle_counts` columns have reduced by two, meaning we have **1019 samples**.

2.2 Reading Metadata for CCLE RNA-seq Samples

Metadata -> Cell lines/Samples x Information

```r
#Path to metadata, tab-separated
metadata <- "/Users/emrunali/bulkRNAseq/Cell_lines_annotations_20181226.txt"

#Reading metadata
ccle_meta <- read.csv(metadata, sep='\t')
```

```r
#Checking dimension of ccle_meta
print(dim(ccle_meta))
```

```
[1] 1461   33
```

2.2.1 Taking intersection of samples in metadata and counts data

We have information for **1461 CCLE samples/cell lines** in this metadata. But our counts data has 1019 (and all unique) samples only. So we will need to look for common samples among both of these data and align them in same order because if rows in metadata and columns in counts data don't match exactly, we risk pairing the wrong biological labels with expression data. This compromises all downstream statistical analysis, biological interpretation, and reproducibility.

```r
#Find common samples among both datasets
common <- intersect(colnames(ccle_counts), ccle_meta$CCLE_ID)

# Subset and reorder count matrix columns and metadata rows so sample order
matches 'common'
ccle_counts <- ccle_counts[, match(common, colnames(ccle_counts))]
ccle_meta   <- ccle_meta[match(common, ccle_meta$CCLE_ID), ]

#Sanity check for same order of samples in both datasets
print(head(ccle_meta$CCLE_ID))
```

```
[1] "A101D_SKIN"                       "A1207_CENTRAL_NERVOUS_SYSTEM"
[3] "A172_CENTRAL_NERVOUS_SYSTEM"   "A204_SOFT_TISSUE"
[5] "A2058_SKIN"                       "A253_SALIVARY_GLAND"
```

```r
print(head(colnames(ccle_counts)))
```

```
[1] "A101D_SKIN"                       "A1207_CENTRAL_NERVOUS_SYSTEM"
[3] "A172_CENTRAL_NERVOUS_SYSTEM"   "A204_SOFT_TISSUE"
[5] "A2058_SKIN"                       "A253_SALIVARY_GLAND"
```

Now, the order of samples in counts and meta data look the same.

```r
# No. of columns/samples in ccle_counts
ncol(ccle_counts)
```

```
[1] 1004
```

```r
#No. of rows/samples in ccle_meta
nrow(ccle_meta)
```

```
[1] 1004
```

So we have **1004 samples** in the dataset after preprocessing so far.

2.2.2 Select useful metadata columns

Currently, metadata has 33 columns and not all are needed for our DE analysis. Keeping only what's relevant (e.g., tissue type, treatment group, lineage) makes the design formulas for DE analysis and the interpretation of results much clearer.

```r
# Sneak peek into content of metadata columns
lapply(ccle_meta, function(x) head(unique(x), 10))  # show first 10 unique
per column
```

```
$CCLE_ID
 [1] "A101D_SKIN"
 [2] "A1207_CENTRAL_NERVOUS_SYSTEM"
 [3] "A172_CENTRAL_NERVOUS_SYSTEM"
 [4] "A204_SOFT_TISSUE"
 [5] "A2058_SKIN"
 [6] "A253_SALIVARY_GLAND"
 [7] "A2780_OVARY"
 [8] "A375_SKIN"
 [9] "A3KAW_HAEMATOPOIETIC_AND_LYMPHOID_TISSUE"
[10] "A427_LUNG"

$depMapID
 [1] "ACH-000008" "ACH-000283" "ACH-000558" "ACH-000201" "ACH-000788"
 [6] "ACH-000740" "ACH-000657" "ACH-000219" "ACH-000697" "ACH-000757"

$Name
 [1] "A101D"  "A1207"  "A172"   "A-204"  "A2058"  "A-253"  "A2780"  "A-375"
 [9] "A3/KAW" NA

$Pathology
[1] "primary"          ""                 "metastasis"       NA
[5] "benign_neoplasia"

$Site_Primary
 [1] "skin"                                 "central_nervous_system"
 [3] "soft_tissue"                          "salivary_gland"
 [5] "ovary"                                "haematopoietic_and_lymphoid_tissue"
 [7] NA                                     "kidney"
 [9] "lung"                                 "bone"

$Site_Subtype1
```

```
 [1] "NS"              "brain"           "striated_muscle" "submaxillary"
 [5] NA                "bladder"         "head_neck"       "mouth"
 [9] "larynx"          "pharynx"

$Site_Subtype2
 [1] "NS"          NA            "tongue"      "sigmoid"     "right"
 [6] "hypopharynx" "kidney"      "uterus"      "abdomen"     "ovary"

$Site_Subtype3
[1] "NS" NA

$Histology
 [1] "malignant_melanoma"
 [2] "glioma"
 [3] "rhabdoid_tumour"
 [4] "carcinoma"
 [5] "lymphoid_neoplasm"
 [6] NA
 [7] "Ewings_sarcoma-peripheral_primitive_neuroectodermal_tumour"
 [8] "mesothelioma"
 [9] "haematopoietic_neoplasm"
[10] "chondrosarcoma"

$Hist_Subtype1
 [1] "NS"
 [2] "astrocytoma_Grade_IV"
 [3] "mucoepidermoid_carcinoma"
 [4] "adenocarcinoma"
 [5] "diffuse_large_B_cell_lymphoma"
 [6] NA
 [7] "renal_cell_carcinoma"
 [8] "non_small_cell_carcinoma"
 [9] "acute_lymphoblastic_T_cell_leukaemia"
[10] "acute_myeloid_leukaemia"

$Hist_Subtype2
 [1] "NS"
 [2] "glioblastoma_multiforme"
 [3] NA
 [4] "M5"
 [5] "papillary_transitional_cell_carcinoma"
 [6] "medullary"
 [7] "papillary"
 [8] "M7"
 [9] "M6"
[10] "M4"

$Hist_Subtype3
[1] "NS" NA

$Gender
[1] "male"   "female" NA       ""

$Life_Stage
[1] ""          "pediatric" NA          "adult"

$Age
 [1] 56 NA 53  1 43 54 52 58 15 78

$Race
 [1] "caucasian"        ""                 NA                 "asian"
 [5] "african_american" "african"          "east_indian"      "american_indian"
 [9] "turkish"          "north_african"

$Geo_Loc
 [1] ""          NA          "chinese"   "taiwan"    "japan"     "france"
 [7] "argentina" "sweden"    "canada"    "europe"

$inferred_ethnicity
[1] "Caucasian"        "African_american" "Asian"            NA

$Site_Of_Finding
 [1] ""            "lymph_node"  NA            "pleura"      "ascites"
 [6] "skin"        "NS"          "liver"       "brain"       "soft_tissue"

$Disease
 [1] ""                      "brain_cancer"         "melanoma"
 [4] "ovarian_cancer"        NA                     "renal_cell_carcinoma"
 [7] "lung_cancer"           "Ewings_sarcoma"       "kidney_cancer"
[10] "TALL"

$Annotation_Source
[1] "CCLE"         "ACHILLES"     NA             "COSMIC"       "collaborator"

$Original.Source.of.Cell.Line
 [1] "GNF"   NA      "ATCC"  "NIBRI" "HSRRB" ""      "JCRB"  "RIKEN" "DSMZ"
```

```
[10] "HSSRB"

$Characteristics
 [1] ""                                NA
 [3] "Adherent"                        "Monolayer and suspension"
 [5] "suspension; grows in aggregates" "adherent"
 [7] "SUSPENSION"                      "adherent; epithelial; DT=22 hrs"
 [9] "Adherent; 0.25% TRYPSIN"         "fibroblast-like"

$Growth.Medium
 [1] ""                     NA                     "DMEM +10%FBS"
 [4] "McCoy's 5A + 10% FBS" "RPMI1640+10%FBS"      "EMEM+10%FBS"
 [7] "RPMI-10% FBS"         "F-12 +10%FBS"         "MEM+10% FBS"
[10] "EMEM+10% FBS"

$Supplements
 [1] ""
 [2] NA
 [3] "2mM L-Glutamine"
 [4] "0.005mg/ml insulin, 0.005mg/ml transferrin and 5ng/ml GM-CSF"
 [5] "0.5%human serum(+0.005"
 [6] "N/A"
 [7] "2mM L-GLUTAMINE + 0.4UG/ML HYDROCORTISONE"
 [8] "2mM L-glutamine, 0.4ug/ml hydrocortisone"
 [9] "2mM GLUTAMINE+0.4UG/ML HYDROCORTISONE"
[10] "2mM L-Glutamine ; 0.4 ug/ml Hydrocortisone"

$Freezing.Medium
 [1] ""            NA            "5%DMSO"     "5% DMSO"    "10%DMSO"
 [6] "10% DMSO"    "?"           "5-7.5%DMSO" "5%DMSo"     "5 % DMSO"

$Doubling.Time.from.Vendor
 [1] ""              NA              "~  48 hrs"    "50-90hrs"     "~  30 hrs"
 [6] "~  40-50 hrs" "~  50hrs"      "~  2-3 days"  "~  70 hrs"    "~ 35 hrs"

$Doubling.Time.Calculated.hrs
 [1]  69.1    NA  67.5  47.3  73.3  30.8  40.0  53.6 118.1  54.0

$type
 [1] "melanoma"       "glioma"         "soft_tissue"    "other"
 [5] "ovary"          "lymphoma_DLBCL" NA               "kidney"
 [9] "lung_NSC"       "Ewings_Sarcoma"
```

```
$type_refined
 [1] "melanoma"             "glioma"               "soft_tissue"
 [4] "upper_aerodigestive" "ovary"                "lymphoma_DLBCL"
 [7] "lung_NSC"             "kidney"               "Ewings_sarcoma"
[10] "mesothelioma"

$PATHOLOGIST_ANNOTATION
 [1] "Skin:Melanoma"                 NA
 [3] "CNS:Glioma_HighGrade"          "Soft_Tissue:Sarcoma_Rhabdoid"
 [5] "Salivary_Gland:Carcinoma"      "Ovary:Carcinoma"
 [7] "Lymphoma:NH_B_cell"            "Kidney:Carcinoma"
 [9] "Lung:NSCLC_Others"             "Bone:Sarcoma_Ewing"

$mutRate
 [1]  78.14513        NA 135.61907  92.67187 164.67689 150.15779 206.54899
 [8] 150.05814 211.42552 108.03838

$tcga_code
 [1] "SKCM"                NA                    "GBM"
 [4] "SARC"                "HNSC"                "UNABLE TO CLASSIFY"
 [7] "DLBC"                "KIRC"                "LUAD"
[10] "MESO"
```

```
#Keeping only useful metadata columns
ccle_meta <- ccle_meta[, c("CCLE_ID","depMapID","Name", "Site_Primary",
"Pathology","Histology","Gender", "Age", "Race", "tcga_code",
"mutRate","Growth.Medium", "Doubling.Time.Calculated.hrs")]


print(head(ccle_meta)[1:3,])
```

```
                        CCLE_ID   depMapID  Name           Site_Primary
585                  A101D_SKIN ACH-000008 A101D                   skin
837 A1207_CENTRAL_NERVOUS_SYSTEM ACH-000283 A1207 central_nervous_system
675  A172_CENTRAL_NERVOUS_SYSTEM ACH-000558  A172 central_nervous_system
    Pathology          Histology Gender Age      Race tcga_code   mutRate
585   primary malignant_melanoma   male  56 caucasian      SKCM  78.14513
837                       glioma female  NA                <NA>        NA
675   primary             glioma   male  53                 GBM 135.61907
    Growth.Medium Doubling.Time.Calculated.hrs
585                                       69.1
837          <NA>                           NA
```

```
675  DMEM +10%FBS                           67.5
```

Although for defining biologically different groups we don't need to keep `Growth.Medium` and `Doubling.Time.Calculated.hrs` but we are keeping them just in case we biological subgroups are unclear or too small/unbalanced and can look into these potential technical variations.

2.2.3 Cleaning metadata entries

Many entries in the metadata are missing, i.e., are *blank* or *NA*. These need to be handled because most analysis frameworks (e.g. DESeq2, limma, regression models) cannot include rows with*NA* in any covariates used in the design formula. If you specify `~ histology + batch` and one of those columns has NAs, those samples will be dropped automatically by the model — possibly reducing your sample size or biasing results without you realizing it.

```r
# Treat "" (blank) as NA so missingness is counted correctly
ccle_meta[ccle_meta == ""] <- NA

# Calculate fraction of non-missing values per column
sapply(ccle_meta, function(x) mean(!is.na(x)))
```

```
                     CCLE_ID                      depMapID
                   1.0000000                     1.0000000
                        Name                  Site_Primary
                   0.9651394                     0.9651394
                   Pathology                     Histology
                   0.9322709                     0.9651394
                      Gender                           Age
                   0.8466135                     0.7440239
                        Race                     tcga_code
                   0.5428287                     0.8794821
                     mutRate                 Growth.Medium
                   0.9452191                     0.6872510
Doubling.Time.Calculated.hrs
                   0.5169323
```

Columns `Race` and `Doubling.Time.Calculated.hrs` have many missing values but we won't drop them because they might still provide context or be used later for exploratory analysis. Re-importing later is harder than keeping them with missing values recoded.

But, we will replace missing values in categorical columns of metadata with *missing-info*.

```r
# Replace NA with "missing-info" only in character columns
ccle_meta[] <- lapply(ccle_meta, function(col) {
  if (is.character(col) || is.factor(col)) {
    col[is.na(col)] <- "missing-info"
    return(as.character(col))  # keep as character
  } else {
    return(col)  # leave numeric columns (Age, mutRate,
    Doubling.Time.Calculated.hrs) untouched
  }
})

print(head(ccle_meta)[1:3,])
```

```
                         CCLE_ID   depMapID  Name           Site_Primary
585                   A101D_SKIN ACH-000008 A101D                   skin
837 A1207_CENTRAL_NERVOUS_SYSTEM ACH-000283 A1207 central_nervous_system
675  A172_CENTRAL_NERVOUS_SYSTEM ACH-000558  A172 central_nervous_system
       Pathology          Histology Gender Age         Race    tcga_code
585      primary malignant_melanoma   male  56    caucasian         SKCM
837 missing-info             glioma female  NA missing-info missing-info
675      primary             glioma   male  53 missing-info          GBM
      mutRate Growth.Medium Doubling.Time.Calculated.hrs
585  78.14513  missing-info                         69.1
837        NA  missing-info                           NA
675 135.61907  DMEM +10%FBS                         67.5
```

As we can see, numeric columns still have NA values intact and we will thus exclude them from design in DESeq2 but can still use for exploratory plots later.

Subset both datasets for a specific cancer

```r
#Check counts per cancer
table(ccle_meta$Site_Primary)
```

```
                 autonomic_ganglia                      biliary_tract
                                16                                  8
                              bone                             breast
                                26                                 57
            central_nervous_system                        endometrium
                                63                                 28
```

```
haematopoietic_and_lymphoid_tissue                            kidney
                               169                                30
                  large_intestine                              liver
                                58                                25
                              lung                      missing-info
                               181                                35
                        oesophagus                             ovary
                                26                                46
                          pancreas                            pleura
                                41                                11
                          prostate                    salivary_gland
                                 7                                 2
                              skin                   small_intestine
                                56                                 1
                       soft_tissue                           stomach
                                20                                36
                           thyroid        upper_aerodigestive_tract
                                 9                                32
                     urinary_tract
                                21
```

I am interested in exploring either of these: large intestine, stomach, ovary or skin.

```r
# Cancers of interest
tissues <- c("large_intestine", "stomach", "ovary", "skin")

# Summarize missingness
missing_summary <- ccle_meta %>%
  filter(Site_Primary %in% tissues) %>%
  group_by(Site_Primary) %>%
  summarise(
    across(
      .cols = everything(),
      .fns = ~ {
        if (is.numeric(.)) {
          # For numeric columns, count NA
          mean(is.na(.)) * 100
        } else {
          # For categorical, count "missing-info"
          mean(. == "missing-info") * 100
        }
      },
      .names = "pct_missing_{col}"
    ),
    .groups = "drop"
  )

print(missing_summary)
```

```
# A tibble: 4 x 13
  Site_Primary    pct_missing_CCLE_ID pct_missing_depMapID pct_missing_Name
  <chr>                         <dbl>                <dbl>            <dbl>
1 large_intestine                   0                    0                0
2 ovary                             0                    0                0
3 skin                              0                    0                0
4 stomach                           0                    0                0
# i 9 more variables: pct_missing_Pathology <dbl>, pct_missing_Histology <dbl>,
#   pct_missing_Gender <dbl>, pct_missing_Age <dbl>, pct_missing_Race <dbl>,
#   pct_missing_tcga_code <dbl>, pct_missing_mutRate <dbl>,
#   pct_missing_Growth.Medium <dbl>,
#   pct_missing_Doubling.Time.Calculated.hrs <dbl>
```

Considering the available data for comparable DE analysis, I selected to go ahead with **large intestine cancer (also known as colon cancer)**. We will also drop `Doubling.Time.Calculated.hrs` since for this cancer type, it is anyway missing ~69% of data.

```
# Subset metadata for large_intestine
ccle_meta_li <- ccle_meta %>%
  filter(Site_Primary == "large_intestine") %>%
  dplyr::select(-Doubling.Time.Calculated.hrs)  # drop the column

# Subset counts matrix for large intestine sampels only
ccle_counts_li <- ccle_counts[, ccle_meta_li$CCLE_ID]

# Sanity check
all(colnames(ccle_counts_li) == ccle_meta_li$CCLE_ID) # Should return TRUE
```

```
[1] TRUE
```

```
dim(ccle_counts_li); dim(ccle_meta_li)
```

```
[1] 17693    58
```

```
[1] 58 12
```

Finally, we have **58 samples** to work with.

2.3 Save outputs

```r
outdir <- "/Users/emrunali/bulkRNAseq/outs"


# Create the directory if it doesn't exist
if (!dir.exists(outdir)) {
  dir.create(outdir, recursive = TRUE)
}

write.csv(ccle_counts, "/Users/emrunali/bulkRNAseq/outs/1_ccle_counts.csv")
write.csv(ccle_meta, "/Users/emrunali/bulkRNAseq/outs/1_ccle_meta.csv",
row.names = FALSE)

write.csv(ccle_counts_li,
"/Users/emrunali/bulkRNAseq/outs/1_ccle_counts_li.csv")
write.csv(ccle_meta_li, "/Users/emrunali/bulkRNAseq/outs/1_ccle_meta_li.csv",
row.names = FALSE)
```

3 Principal Component Analysis (PCA)

PCA is an **exploratory diagnostic step** that helps us:

- Understand the main sources of variation in your RNA-seq data.
- Decide on appropriate design covariates.
- Detect outliers or batch effects early.
- Gain confidence that your data makes biological sense before moving to DESeq2/edgeR modeling.

But before that, we need to obtain variance-stabilized counts to reduce heteroscedasticity (variance of counts grows with the mean, i.e., highly expressed genes dominate the variance, while low-expressed genes are drowned out). This way PCA (and clustering/heatmaps) reflects real biological structure rather than technical noise.

3.1 Variance stabilizing transformation (`vst()`)

```r
# Set rownames of metadata to CCLE_ID so they match colnames of counts
rownames(ccle_meta_li) <- ccle_meta_li$CCLE_ID

# Create DESeqDataSet object
dds <- DESeqDataSetFromMatrix(countData = data.frame(ccle_counts_li),
                              colData = ccle_meta_li,
                              design = ~ Pathology)
```

```
converting counts to integer mode
```

```
Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
design formula are characters, converting to factors
```

```r
# Keep  genes with total count >= 100 only
keep_genes <- rowSums(counts(dds)) >= 100

# Subset the DESeqDataSet to retain only those genes
dds <- dds[keep_genes, ]
```

```r
# Variance stabilizing transformation
vsd <- vst(dds)
```

```r
# Verify the vst transformation
print(head(data.frame(counts(dds)), 2))
```

```
       C2BBE1_LARGE_INTESTINE CCK81_LARGE_INTESTINE CL11_LARGE_INTESTINE
OR4F5                       4                     3                    5
OR4F29                      0                     2                    4
       CL14_LARGE_INTESTINE CL34_LARGE_INTESTINE CL40_LARGE_INTESTINE
OR4F5                     7                   13                   10
OR4F29                    4                    8                   12
       COLO201_LARGE_INTESTINE COLO320_LARGE_INTESTINE COLO678_LARGE_INTESTINE
OR4F5                        7                       7                       4
OR4F29                       4                       7                       2
       CW2_LARGE_INTESTINE GP2D_LARGE_INTESTINE HCC56_LARGE_INTESTINE
OR4F5                    3                    3                     6
```

```
OR4F29                     3                     2                     3
       HCT116_LARGE_INTESTINE HCT15_LARGE_INTESTINE HS255T_FIBROBLAST
OR4F5                       6                     6                 4
OR4F29                      1                     6                 2
       HS675T_FIBROBLAST HS698T_FIBROBLAST HT115_LARGE_INTESTINE
OR4F5                  3                 2                     6
OR4F29                 0                 1                    11
       HT29_LARGE_INTESTINE HT55_LARGE_INTESTINE KM12_LARGE_INTESTINE
OR4F5                     0                    1                    1
OR4F29                    3                    1                    1
       LOVO_LARGE_INTESTINE LS1034_LARGE_INTESTINE LS123_LARGE_INTESTINE
OR4F5                     3                      0                     2
OR4F29                    0                      2                     2
       LS180_LARGE_INTESTINE LS411N_LARGE_INTESTINE LS513_LARGE_INTESTINE
OR4F5                      2                      2                     4
OR4F29                     2                      6                     5
       MDST8_LARGE_INTESTINE NCIH508_LARGE_INTESTINE NCIH716_LARGE_INTESTINE
OR4F5                      3                       5                       2
OR4F29                     1                       5                       2
       NCIH747_LARGE_INTESTINE OUMS23_LARGE_INTESTINE RCM1_LARGE_INTESTINE
OR4F5                        2                      3                    8
OR4F29                       6                     10                    6
       RKO_LARGE_INTESTINE SKCO1_LARGE_INTESTINE SNU1033_LARGE_INTESTINE
OR4F5                    0                     2                      10
OR4F29                   2                     2                       7
       SNU1040_LARGE_INTESTINE SNU1197_LARGE_INTESTINE SNU175_LARGE_INTESTINE
OR4F5                        4                       7                      2
OR4F29                       2                       5                      3
       SNU283_LARGE_INTESTINE SNU407_LARGE_INTESTINE SNU503_LARGE_INTESTINE
OR4F5                       2                      1                      4
OR4F29                      2                      6                      3
       SNU61_LARGE_INTESTINE SNU81_LARGE_INTESTINE SNUC1_LARGE_INTESTINE
OR4F5                      6                     6                     1
OR4F29                     3                     9                     2
       SNUC2A_LARGE_INTESTINE SNUC4_LARGE_INTESTINE SNUC5_LARGE_INTESTINE
OR4F5                       3                    24                    12
OR4F29                      2                     7                    11
       SW1116_LARGE_INTESTINE SW1417_LARGE_INTESTINE SW1463_LARGE_INTESTINE
OR4F5                      27                      0                     27
OR4F29                     27                      1                     14
       SW403_LARGE_INTESTINE SW480_LARGE_INTESTINE SW48_LARGE_INTESTINE
OR4F5                     14                     4                   12
OR4F29                     7                     3                   10
```

```
       SW620_LARGE_INTESTINE SW837_LARGE_INTESTINE SW948_LARGE_INTESTINE
OR4F5                      3                     8                     1
OR4F29                     1                     8                     3
       T84_LARGE_INTESTINE
OR4F5                   10
OR4F29                   7
```

```
print(head(data.frame(assay(vsd)), 2))
```

```
       C2BBE1_LARGE_INTESTINE CCK81_LARGE_INTESTINE CL11_LARGE_INTESTINE
OR4F5                6.090748              6.047946             6.117256
OR4F29               5.723067              5.988516             6.075859
       CL14_LARGE_INTESTINE CL34_LARGE_INTESTINE CL40_LARGE_INTESTINE
OR4F5              6.240252             6.310797             6.306689
OR4F29             6.114914             6.185335             6.361538
       COLO201_LARGE_INTESTINE COLO320_LARGE_INTESTINE COLO678_LARGE_INTESTINE
OR4F5                 6.191285                6.187854                6.060950
OR4F29                6.077669                6.187854                5.962259
       CW2_LARGE_INTESTINE GP2D_LARGE_INTESTINE HCC56_LARGE_INTESTINE
OR4F5             6.018533             6.027700              6.189030
OR4F29            6.018533             5.971953              6.053266
       HCT116_LARGE_INTESTINE HCT15_LARGE_INTESTINE HS255T_FIBROBLAST
OR4F5                6.164534              6.161106          6.151900
OR4F29               5.903881              6.161106          6.026854
       HS675T_FIBROBLAST HS698T_FIBROBLAST HT115_LARGE_INTESTINE
OR4F5           6.061495          6.053909              6.047044
OR4F29          5.723067          5.957263              6.160973
       HT29_LARGE_INTESTINE HT55_LARGE_INTESTINE KM12_LARGE_INTESTINE
OR4F5              5.723067              5.91838             5.917692
OR4F29             6.130586              5.91838             5.917692
       LOVO_LARGE_INTESTINE LS1034_LARGE_INTESTINE LS123_LARGE_INTESTINE
OR4F5              6.013456               5.723067              6.044072
OR4F29             5.723067               6.011952              6.044072
       LS180_LARGE_INTESTINE LS411N_LARGE_INTESTINE LS513_LARGE_INTESTINE
OR4F5               5.994025               5.982093              6.104948
OR4F29              5.994025               6.170521              6.149714
       MDST8_LARGE_INTESTINE NCIH508_LARGE_INTESTINE NCIH716_LARGE_INTESTINE
OR4F5               6.094456                 6.19263                5.998218
OR4F29              5.937883                 6.19263                5.998218
       NCIH747_LARGE_INTESTINE OUMS23_LARGE_INTESTINE RCM1_LARGE_INTESTINE
OR4F5                 6.032412               6.019863             6.254293
OR4F29                6.256846               6.262743             6.183766
```

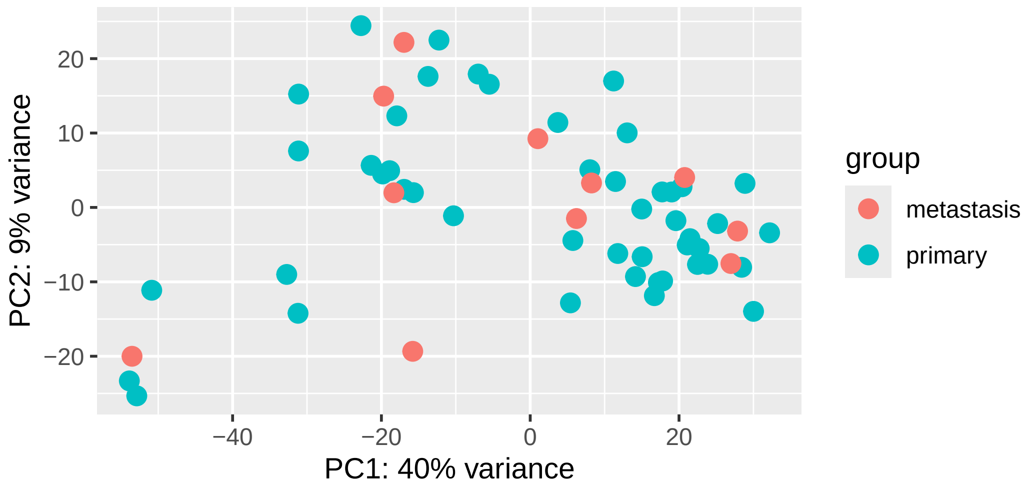
```
       RKO_LARGE_INTESTINE SKCO1_LARGE_INTESTINE SNU1033_LARGE_INTESTINE
OR4F5             5.723067              6.013893                6.297060
OR4F29            6.044745              6.013893                6.204244
       SNU1040_LARGE_INTESTINE SNU1197_LARGE_INTESTINE SNU175_LARGE_INTESTINE
OR4F5                 6.039327                6.194367               5.982438
OR4F29                5.946920                6.121891               6.040518
       SNU283_LARGE_INTESTINE SNU407_LARGE_INTESTINE SNU503_LARGE_INTESTINE
OR4F5                 5.99223               5.903764               6.097168
OR4F29                5.99223               6.164249               6.047274
       SNU61_LARGE_INTESTINE SNU81_LARGE_INTESTINE SNUC1_LARGE_INTESTINE
OR4F5               6.154902              6.328211              5.918342
OR4F29              6.028988              6.461570              5.999018
       SNUC2A_LARGE_INTESTINE SNUC4_LARGE_INTESTINE SNUC5_LARGE_INTESTINE
OR4F5                6.023603              6.619083              6.601198
OR4F29               5.968601              6.212436              6.564860
       SW1116_LARGE_INTESTINE SW1417_LARGE_INTESTINE SW1463_LARGE_INTESTINE
OR4F5                 7.12445               5.723067               7.200289
OR4F29                7.12445               5.940955               6.808115
       SW403_LARGE_INTESTINE SW480_LARGE_INTESTINE SW48_LARGE_INTESTINE
OR4F5               6.660832              6.136331             6.649766
OR4F29              6.391897              6.081268             6.571369
       SW620_LARGE_INTESTINE SW837_LARGE_INTESTINE SW948_LARGE_INTESTINE
OR4F5               6.036035              6.399284              5.924655
OR4F29              5.903995              6.399284              6.071663
       T84_LARGE_INTESTINE
OR4F5             6.201881
OR4F29            6.124219
```

As we can see, the raw counts have been transformed and stored in `vsd` object.

3.2 PCA with `plotPCA()`

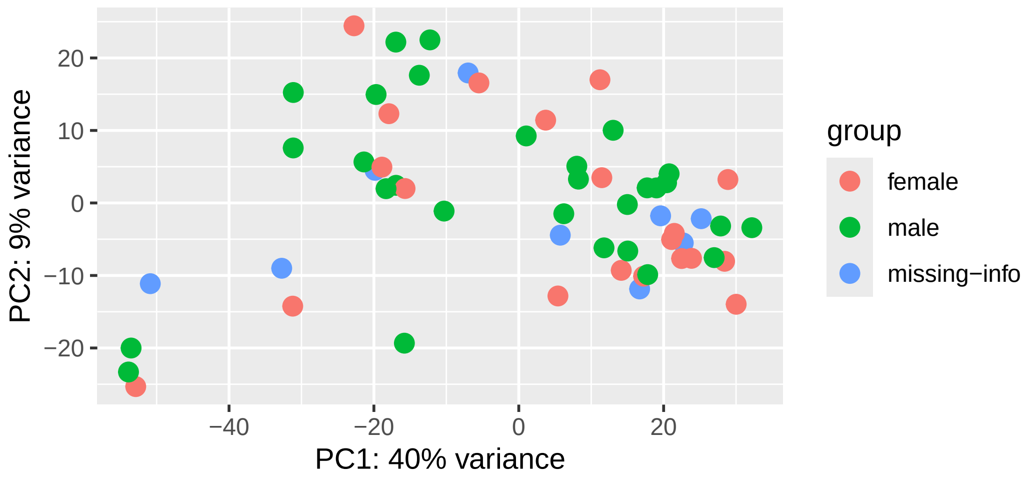
```r
# PCA using top 150 most variable genes, coloring samples by 'Pathology'
plotPCA(vsd, intgroup = "Pathology", ntop = 150)
```

```
using ntop=150 top features by variance
```



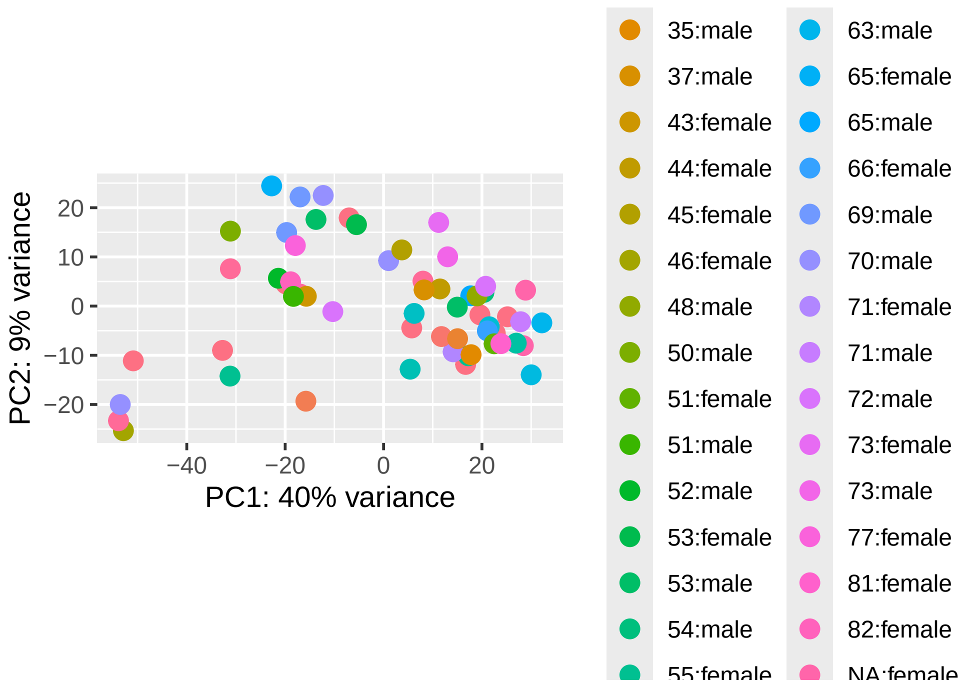
```r
plotPCA(vsd, intgroup = "Gender", ntop = 150)
```

```
using ntop=150 top features by variance
```



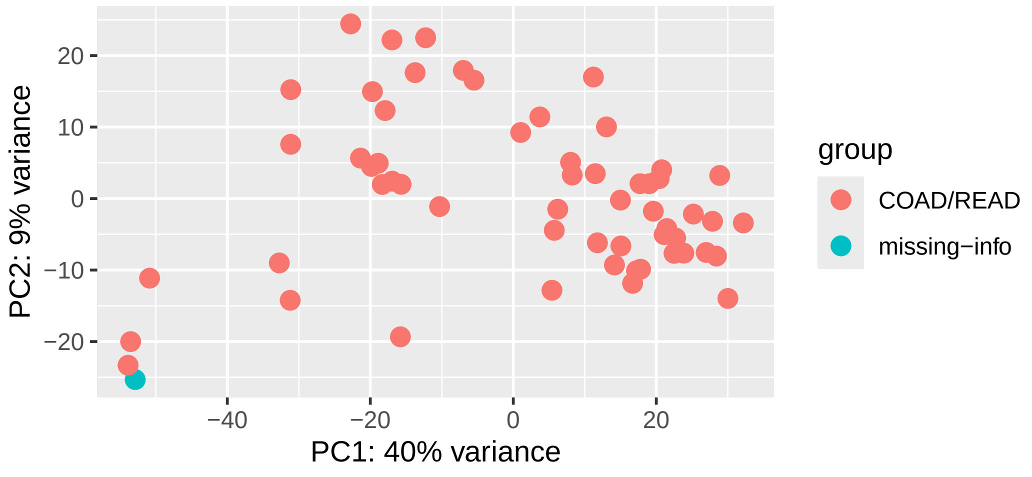
```r
plotPCA(vsd, intgroup = c("Age", "Gender"), ntop = 150)
```

```
using ntop=150 top features by variance
```



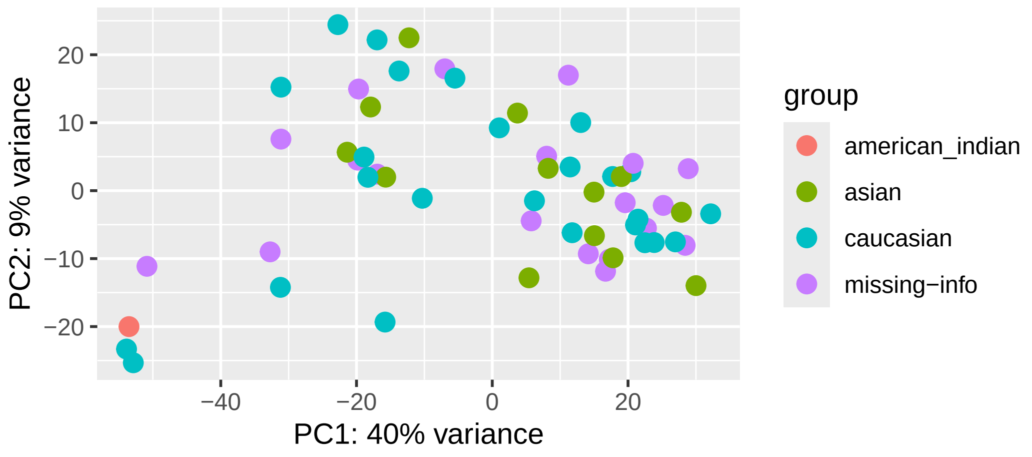
```r
plotPCA(vsd, intgroup = "tcga_code", ntop = 150)
```

```
using ntop=150 top features by variance
```



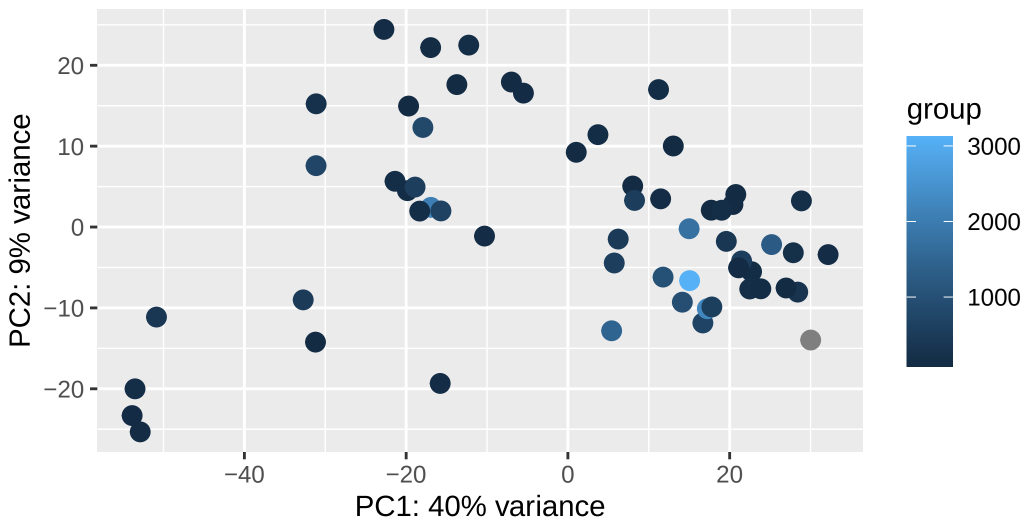
```r
plotPCA(vsd, intgroup = "Race", ntop = 150)
```

```
using ntop=150 top features by variance
```



```
plotPCA(vsd, intgroup = "mutRate", ntop = 150)
```

```
using ntop=150 top features by variance
```



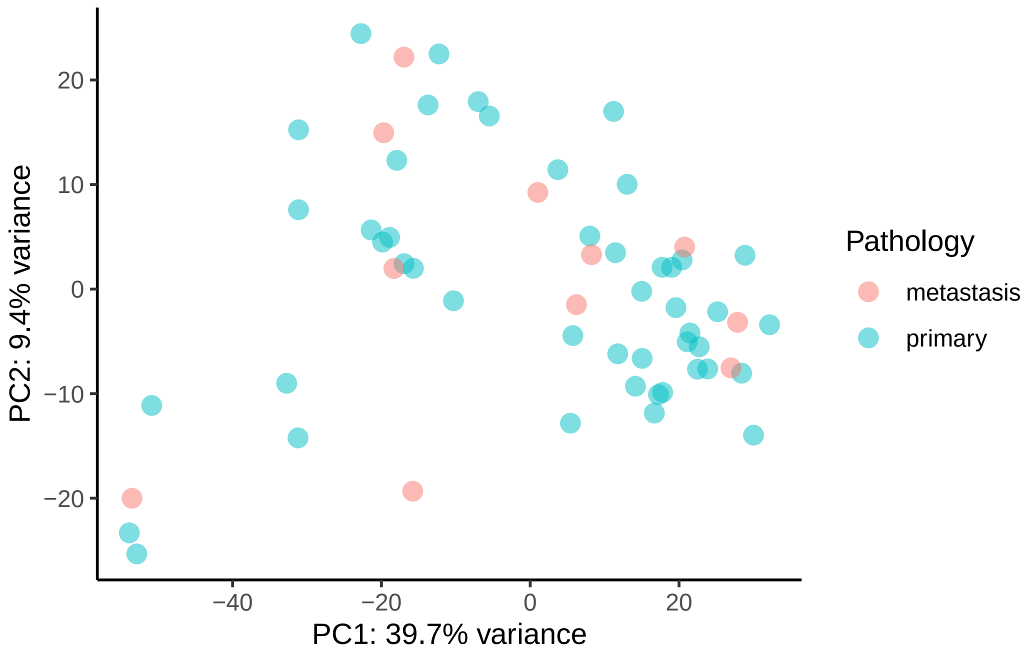
None of these metadata columns seem to be driving the segregation of large-intestine cancer cell lines.

3.3 PCA with `prcomp()`

For deeper exploration of the sources of variation captured by PC1, PC2, and subsequent PCs, we will now perform PCA using `prcomp()` which allows extraction of all PC scores, loadings, and variance explained.

```
# Obtaining the variances of each gene (rows) from vsd matrix
variances <- rowVars(assay(vsd))

#Selecting top 150 most variable genes based on variances obtained above
top_var_genes <- order(variances, decreasing = TRUE)[1:150]

# Subsetting the expression matrix (vsd) to include only the top 150 variable
genes
```

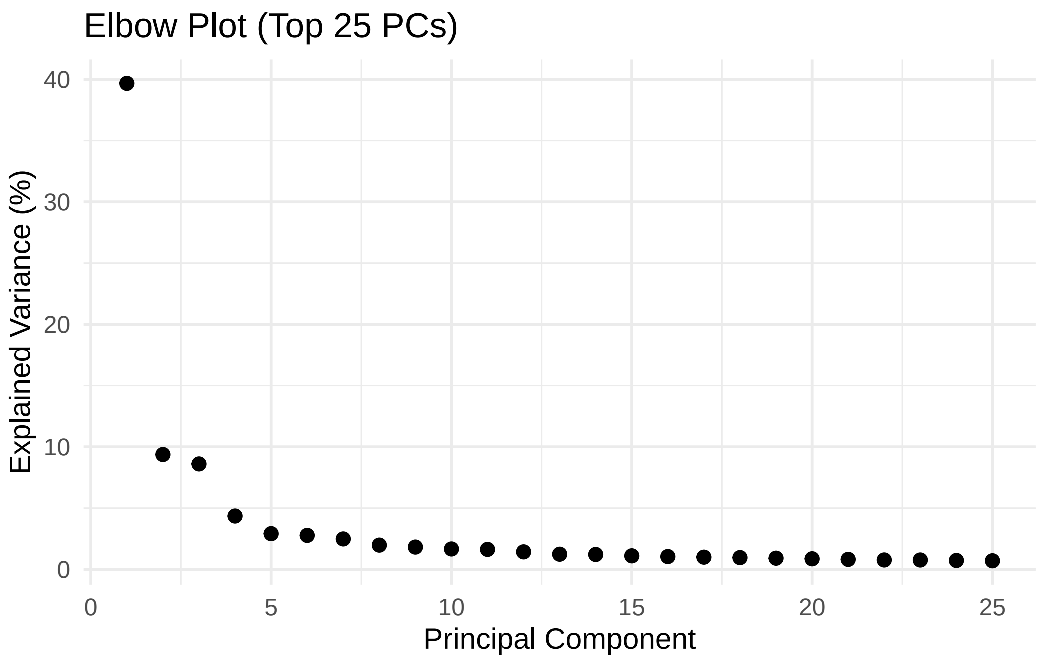
```r
high_var_mat <- assay(vsd)[top_var_genes,]

#Performing PCA using prcomp() to obtain the principal components (PCs) and
variances for top 150 variable genes
pca_res <- prcomp(t(high_var_mat), center = TRUE, scale = FALSE)

#Explanation of the above line in detail:
#transposed matrix with t() as PCA expects samples x features/gene matrix
# center = TRUE centers the data, i.e., subtracts column (gene counts) mean
from each value in the respective column
#prcomp() performs PCA

#Computing the percent variance explained by each PC
explained_variance <- pca_res$sdev^2 / sum(pca_res$sdev^2) * 100

#Creating a combined dataframe for plotting: PC scores of PC1 and PC2 +
metadata
df_pca <- data.frame(PC1 = pca_res$x[, 1], PC2 = pca_res$x[, 2],
ccle_meta_li)

#Creating a scatterplot of PC1 vs. PC2 where each point is colored by
'Pathology' from metadata
ggplot(df_pca, aes(x = PC1, y = PC2, color = Pathology)) +
  geom_point(size = 3, alpha = 0.5) +
  theme_classic() +
  labs(x = paste0("PC1: ", round(explained_variance[1], 1), "% variance"),
       y = paste0("PC2: ", round(explained_variance[2], 1), "% variance"))
```



3.4 Elbow plot

Next, we create an elbow plot to decide how many PCs explain most of the meaningful variance and to avoid including PCs dominated by noise.

```
# Creating dataframe with columns PC and Variance listing first 25 values
from explained_variance which contains percent variance explained by PCs
df_variance <- data.frame(PC = seq_len(25), Variance =
explained_variance[1:25])

# Plotting elbow plot using the above dataframe
ggplot(df_variance, aes(x = PC, y = Variance)) +
  geom_point(size = 2) +
  theme_minimal() +
  labs(title = "Elbow Plot (Top 25 PCs)", x = "Principal Component", y =
  "Explained Variance (%)")
```



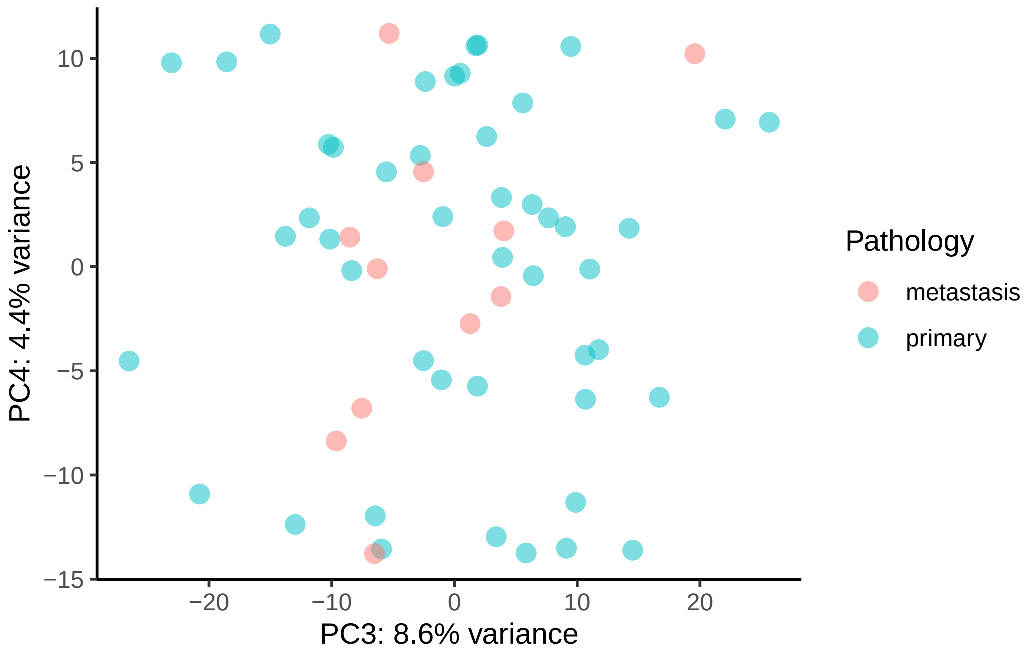
Interpretation:

- The first 2–4 PCs capture most of the meaningful structure in your dataset.
- PCs beyond ~5 mainly represent minor variation or noise.

Thus, for exploratory analysis (PCA plots, clustering), focusing on **PC1–PC4** makes sense.

We already plotted PC1 vs PC2, now let's explore how samples group in PC3 vs PC4 PCA plot.

```
#Creating a combined dataframe for plotting: PC scores of PC3 and PC4 +
metadata
df_pca_34 <- data.frame(PC3 = pca_res$x[, 3], PC4 = pca_res$x[, 4],
ccle_meta_li)

#Creating a scatterplot of PC3 vs. PC4 where each point is colored by
'Pathology' from metadata
ggplot(df_pca_34, aes(x = PC3, y = PC4, color = Pathology)) +
  geom_point(size = 3, alpha = 0.5) +
  theme_classic() +
  labs(x = paste0("PC3: ", round(explained_variance[3], 1), "% variance"),
       y = paste0("PC4: ", round(explained_variance[4], 1), "% variance"))
```



Even PC3 and PC4 capture some variation in the dataset, but this variation does not correspond to strong grouping by *Pathology* status.

3.5 Gene loadings analysis

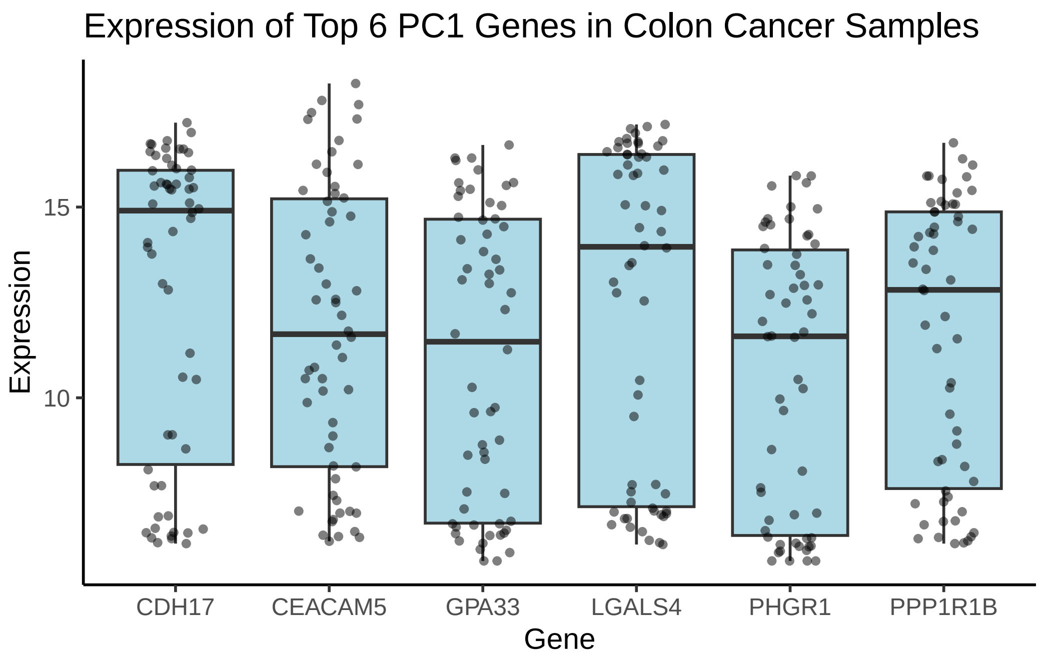
We know that the main drivers of variation are not captured by our current metadata. So we should explore **gene loadings** (weights or coefficients that indicate how much each gene contributes to each PC) which can help us interpret what biological processes or technical factors drive each PC. Basically, this will help us understand *why* our samples separate the way they do in PCA and whether that separation is biologically meaningful or due to unwanted technical artifacts.

```
# Extract the top 6 gene loadings for PC1.
pc1_loadings <- pca_res$rotation[, 1] #pca_res$rotation has gene loadings-
matrix of genes x PCs <=> 150 x 58 (=no. of samples)
pc1_sorted_loadings <- sort(pc1_loadings, decreasing = TRUE)
head(pc1_sorted_loadings, n = 6)
```

```
   LGALS4     CDH17     GPA33   PPP1R1B     PHGR1   CEACAM5
0.1659467 0.1465459 0.1463983 0.1381959 0.1380048 0.1363064
```

Now, we will plot the variance-stabilized counts (from `assay(vsd)`) of these 6 genes across colon cancer cell line samples.

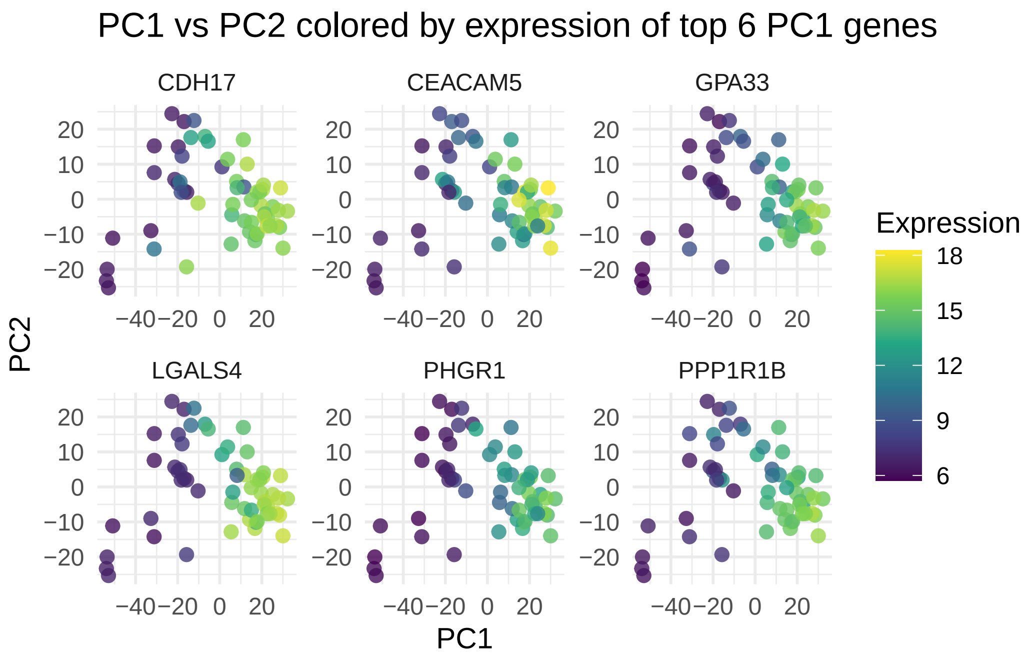
```r
top6_genes <- names(pc1_sorted_loadings)[1:6]

expr_top6 <- assay(vsd)[top6_genes, ]

df_long <- as.data.frame(expr_top6) %>%
  mutate(Gene = rownames(.)) %>%
  pivot_longer(cols = -Gene, names_to = "Sample", values_to = "Expression")

ggplot(df_long, aes(x = Gene, y = Expression)) +
  geom_boxplot(outlier.shape = NA, fill = "lightblue") +
  geom_jitter(width = 0.2, alpha = 0.5, size = 1) +
  theme_classic() +
  labs(title = "Expression of Top 6 PC1 Genes in Colon Cancer Samples")
```



Interpretation:
All six genes show a broad range of expression across samples. Genes **CDH17** and **LGALS4** have relatively high median expression and several samples with high outlier values.

```r
df_long$Sample <- as.character(df_long$Sample)
df_pca$Sample <- rownames(df_pca)
```

```r
df_plot <- left_join(df_long, df_pca, by = "Sample")

ggplot(df_plot, aes(x = PC1, y = PC2, color = Expression)) +
  geom_point(size = 2, alpha = 0.8) +
  facet_wrap(~ Gene, scales = "free") +
  scale_color_viridis_c() +
  theme_minimal() +
  labs(title = "PC1 vs PC2 colored by expression of top 6 PC1 genes", color =
  "Expression")
```



Interpretation:

- Across all six genes **higher expression (yellow-green)** tends to occur on the **right-hand side (positive PC1)**.
- **Lower expression (purple)** is concentrated on the **left-hand side (negative PC1)**.
- This pattern confirms that PC1 is strongly influenced by these genes, consistent with them being the top PC1 loadings.

Biological interpretation:

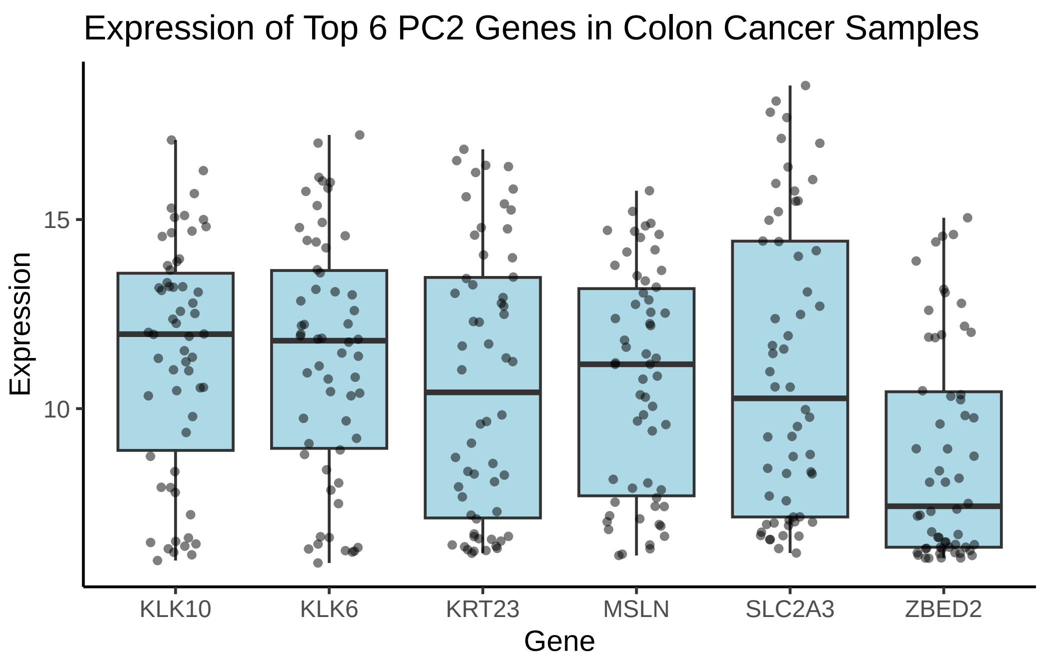
- These six genes collectively maintain intestinal epithelial cell identity, adhesion, differentiation, and proper cellular signaling in the normal colon epithelium.
- Together, their expression pattern defines colorectal tumor differentiation status - high expression indicates metabolically active, differentiated tumors while low expression indicates dedifferentiated, disrupted tumors with distinct therapeutic vulnerabilities.
- **PC1 likely captures this Epithelial Differentiation Status**:
  - **Right side** (high expression):
    - **Preserved intestinal identity** (GPA33, some CDH17 function)
    - **High metabolic/proliferative activity** (active oncogenes: CDH17, CEACAM5, PPP1R1B, PHGR1)
    - **Aggressive through proliferation and adhesion-mediated mechanisms**
  - **Left side** (low expression):
    - **Loss of intestinal differentiation markers** (GPA33, CEACAM5 loss)
    - **Disrupted cellular metabolism** and transport systems
    - **Aggressive through immune evasion and hypermutation**

Similarly, let's explore PC2.

```r
# Extract gene loadings for PC2
pc2_loadings <- pca_res$rotation[, 2]
pc2_sorted_loadings <- sort(pc2_loadings, decreasing = TRUE)
head(pc2_sorted_loadings, n = 6)
```

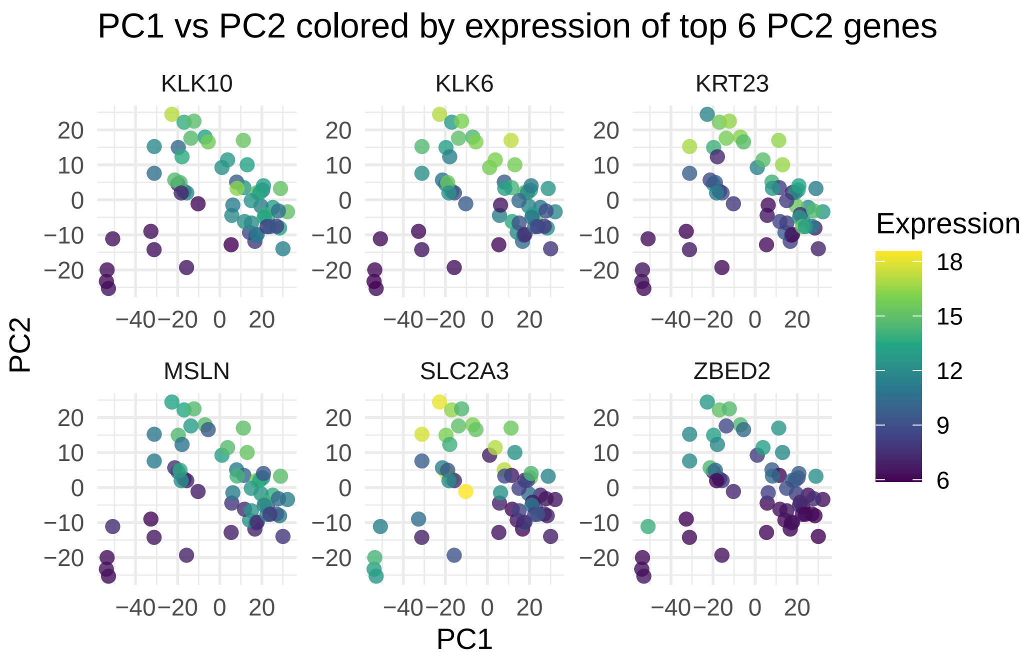
```
     KLK6     KRT23     ZBED2    SLC2A3     KLK10      MSLN 
0.2226096 0.2092485 0.1861759 0.1848876 0.1792647 0.1688836 
```

```r
top6_genes_pc2 <- names(pc2_sorted_loadings)[1:6]
#print(top6_genes)

expr_top6_pc2 <- assay(vsd)[top6_genes_pc2, ]

df_long_pc2 <- as.data.frame(expr_top6_pc2) %>%
  mutate(Gene = rownames(.)) %>%
  pivot_longer(cols = -Gene, names_to = "Sample", values_to = "Expression")

ggplot(df_long_pc2, aes(x = Gene, y = Expression)) +
  geom_boxplot(outlier.shape = NA, fill = "lightblue") +
  geom_jitter(width = 0.2, alpha = 0.5, size = 1) +
  theme_classic() +
  labs(title = "Expression of Top 6 PC2 Genes in Colon Cancer Samples")
```



Interpretation:
Similar to above findings, all six genes that are highly variable along PC2 show a broad range of expression across samples.

```
# Creating faceted PC1-vs-PC2 scatter plots as above
df_long_pc2$Sample <- as.character(df_long_pc2$Sample)
df_pca$Sample <- rownames(df_pca)

df_plot <- left_join(df_long_pc2, df_pca, by = "Sample")

ggplot(df_plot, aes(x = PC1, y = PC2, color = Expression)) +
  geom_point(size = 2, alpha = 0.8) +
  facet_wrap(~ Gene, scales = "free") +
  scale_color_viridis_c() +
  theme_minimal() +
  labs(title = "PC1 vs PC2 colored by expression of top 6 PC2 genes", color =
  "Expression")
```



<u>Interpretation:</u>

- For most genes, **higher expression (yellow-green)** tends to occur on the **upper side (positive PC2)**.
- **Lower expression (purple)** appears on the **lower side (negative PC2)**.
- This pattern confirms that PC2 is influenced by these genes, consistent with them being the top PC2 loadings.

<u>Biological interpretation:</u>

- These six genes collectively regulate proteolytic activity, epithelial structure, metabolic transport, and transcriptional control in normal intestinal cells.
- Together, their expression pattern defines tumor aggressiveness and metabolic reprogramming - high expression indicates highly aggressive, metabolically hyperactive tumors with enhanced invasion capabilities, while low expression indicates less metabolically active tumors with reduced invasive potential.
- **PC2 likely captures "Tumor Aggressiveness and Metabolic Reprogramming Spectrum"**:
  - **Upper region** (high expression):
    - **Enhanced Proteolytic Activity** (KLK10 and KLK6)

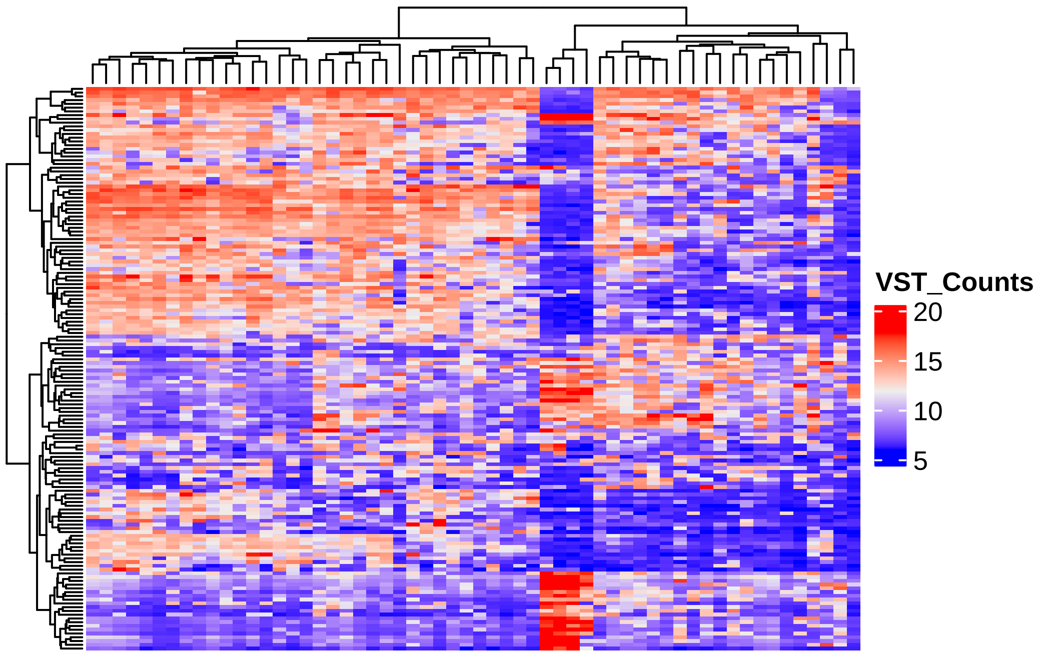
    * **Metabolic Hyperactivity** (SLC2A3 (GLUT3))
    * **Invasive Epithelial Phenotype** (KRT23)
- **Lower region** (low expression):
    * **Reduced Proteolytic Activity** (Lower matrix degradation capability, reduced invasive potential)
    * **Metabolic Constraints** (Limited glucose uptake capacity, reduced nucleotide synthesis, Less adapted to glucose-poor tumor microenvironments)
    * **Less Aggressive Epithelial State** (Reduced EMT signaling and migration capacity, lower immune evasion capabilities)

4 Hierarchical Clustering

Hierarchical clustering is an important next step after PCA as it groups samples based on overall similarity across all variables, helping detect natural subgroups, co-expression patterns, and outliers, and is essential for meaningful heatmap visualization and exploratory QC.

We will select the 150 most variable genes from your variance-stabilized data (`vsd`) and create a hierarchically clustered heatmap.

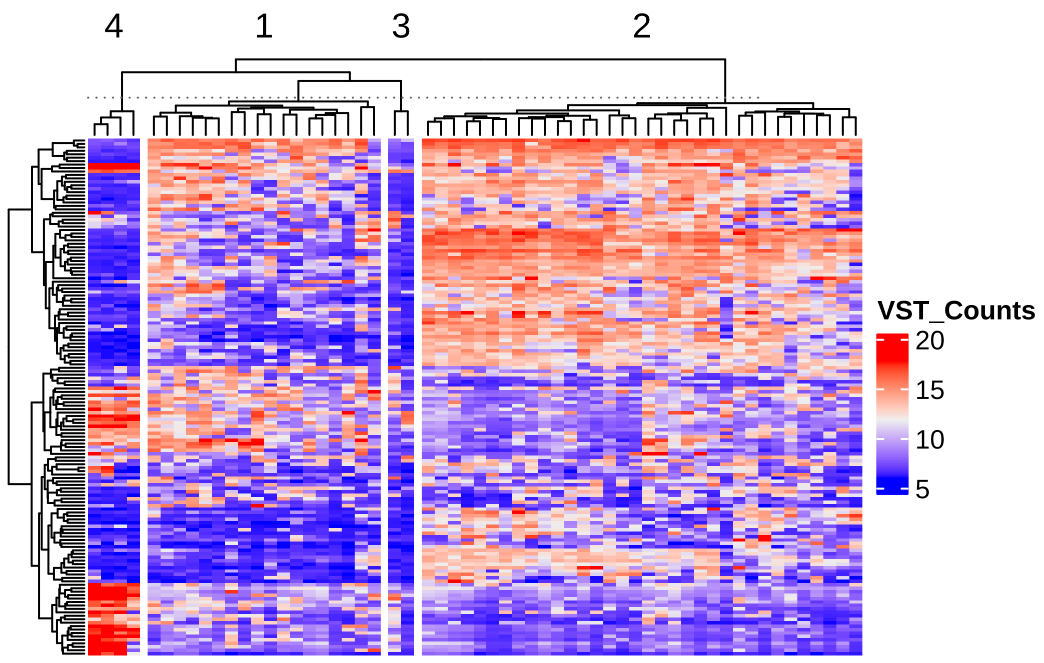
```r
# Identify the top 150 most variable genes across all samples
high_var_genes <- head(order(rowVars(assay(vsd)), decreasing = TRUE), 150)

# Extract the variance-stabilized expression matrix for those top variable
genes
high_var_mat <- assay(vsd)[high_var_genes,]

# Generate a heatmap of the top 150 most variable genes
Heatmap(high_var_mat,
        clustering_distance_columns = "euclidean",
        clustering_method_columns = "complete",
        clustering_distance_rows = "euclidean",
        clustering_method_rows = "complete",
        show_row_names = FALSE,
        show_column_names = FALSE,
        name = "VST_Counts",)
```



It looks like we can create 4 clusters from this.

4.1 Dividing samples into four clusters

```
# Cutting the dendrogram above into 4 cluster
hc <- hclust(dist(t(high_var_mat)), method = "complete")
clusters <- cutree(hc, k = 4)

# Heatmap generation
Heatmap(high_var_mat,
        clustering_distance_columns = "euclidean",
        clustering_method_columns = "complete",
        clustering_distance_rows = "euclidean",
        clustering_method_rows = "complete",
        show_row_names = FALSE,
        show_column_names = FALSE,
        name = "VST_Counts",
        column_split = clusters)
```



The resulting 4 clusters look well-defined.

We could also check whether anything in the metadata corresponds to these clusters.

```r
# Getting list of values in columns of interest to color-code them later
unique(ccle_meta_li$Pathology)
```

```
[1] "primary"    "metastasis"
```

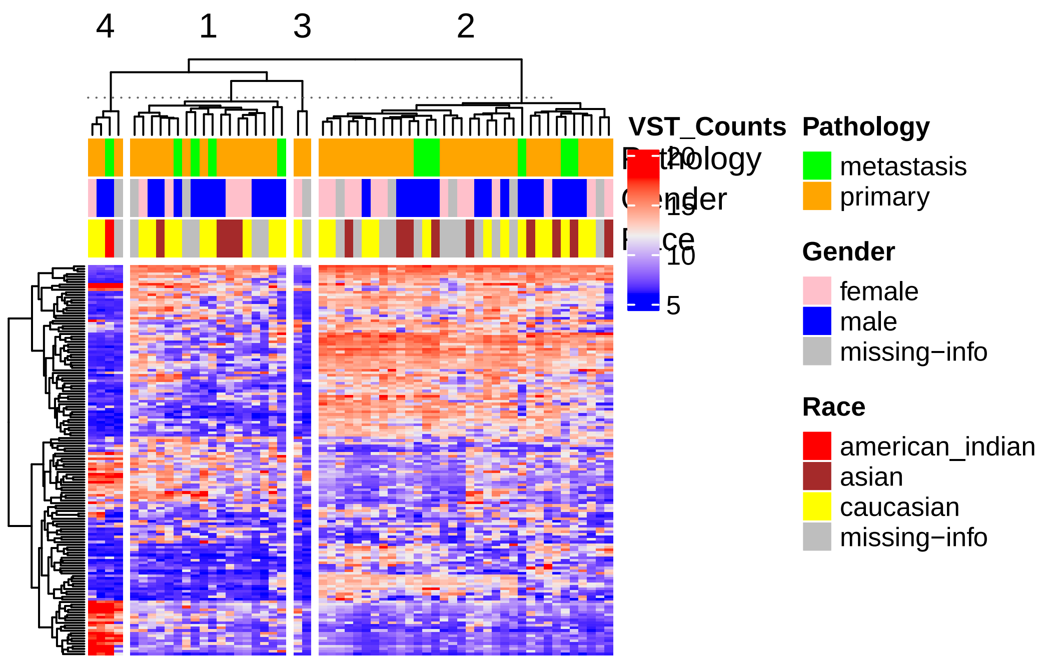
```r
unique(ccle_meta_li$Gender)
```

```
[1] "male"         "missing-info" "female"
```

```r
unique(ccle_meta_li$Race)
```

```
[1] "caucasian"       "missing-info"    "american_indian" "asian"
```

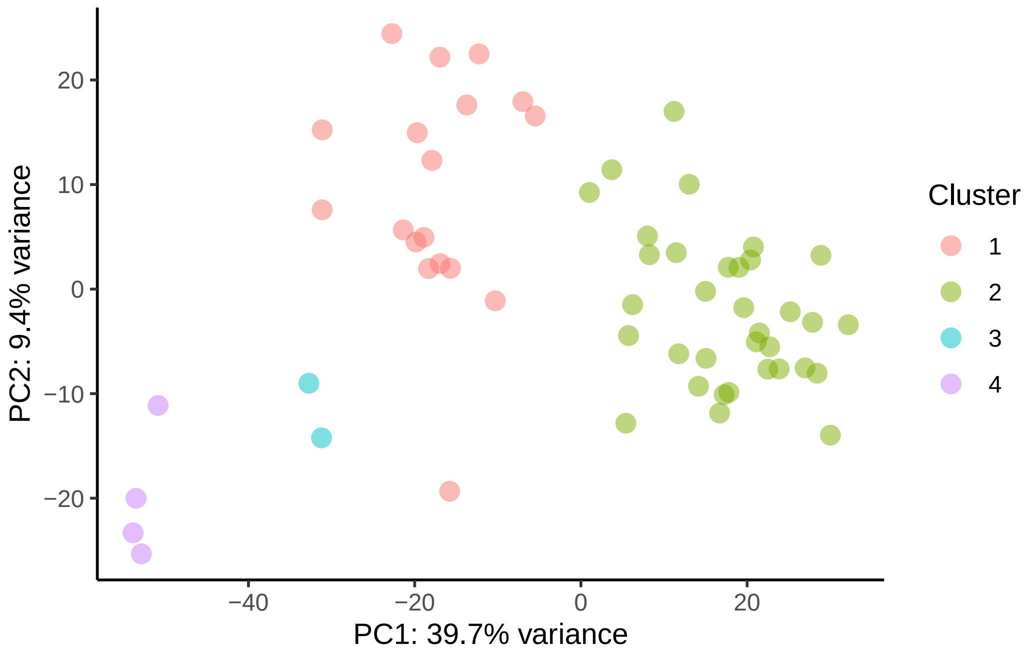
```r
# Copy metadata for convenience, then keep only the columns you want to
display above the heatmap
sample_annotation_df <- ccle_meta_li
sample_annotation_df <- sample_annotation_df %>% dplyr::select(Pathology,
Gender, Race)

# Define a color map for each annotation column (values → colors)
sample_annotation_colors <- list(
  Pathology = c("primary" = "orange", "metastasis" = "green"),
  Gender = c("male" = "blue", "female" = "pink", "missing-info" = "grey"),
  Race = c("caucasian" = "yellow", "american_indian" = "red", "asian" =
  "brown", "missing-info" = "grey")
)

# Build a ComplexHeatmap annotation object using the data frame and color
maps
# 'annotation_label' sets the displayed labels for the annotation tracks
column_annotation <- HeatmapAnnotation(
  df = sample_annotation_df,
  col = sample_annotation_colors,
  annotation_label = c("Pathology", "Gender", "Race")
)

# Draw the heatmap of the top variable genes with hierarchical clustering
# 'top_annotation' adds the colored bars built above to the top of the
heatmap
Heatmap(high_var_mat,
        clustering_distance_columns = "euclidean",
        clustering_method_columns = "complete",
        clustering_distance_rows = "euclidean",
        clustering_method_rows = "complete",
        show_row_names = FALSE,
        show_column_names = FALSE,
        name = "VST_Counts",
        column_split = clusters,
        top_annotation = column_annotation)
```



No clear separation of clusters can be attributed to either pathology, gender or race. `tcga_code` is the same across all samples and other metadata columns are numeric.

So, we can look into how these clusters defined via hierarchical clustering correspond to what we see in the PCA plot.

```
#Creating a PCA dataframe PC scores of PC1 and PC2 for plotting
pca_df <- data.frame(pca_res$x[, 1:2])

#Clustering the dataset into 2 groups for simplicity
#clusters <- cutree(hc, k = 2)

# Adding cluster assignments to PCA dataframe
pca_df$Cluster <- factor(clusters)

# Creating a scatterplot of PC1 vs. PC2 where each point is colored by
'Cluster'
ggplot(pca_df, aes(x = PC1, y = PC2, color = Cluster)) +
  geom_point(size = 3, alpha = 0.5) +
  scale_color_manual(values = hue_pal()(length(unique(clusters)))) +
  theme_classic() +
  labs(x = paste0("PC1: ", round(explained_variance[1], 1), "% variance"),
       y = paste0("PC2: ", round(explained_variance[2], 1), "% variance"),
       color = "Cluster")
```

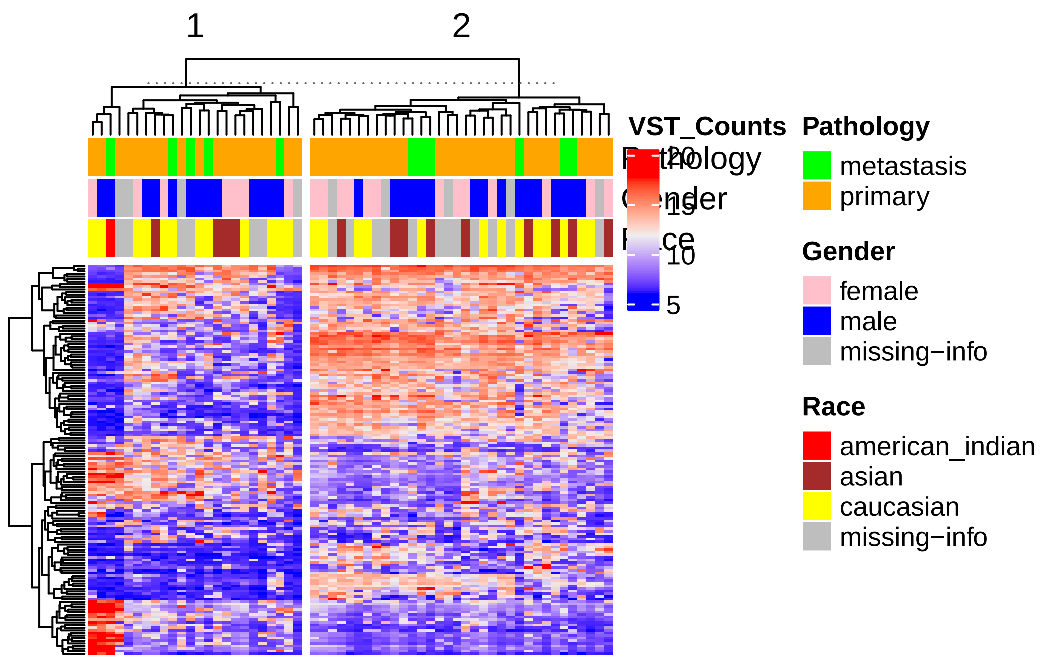


The above plot reflects that the 4 clusters defined by hierarchical clustering also show up in the PCA plot as distinct groups.

But clusters 3 and 4 have very few samples, so for simplicity of DE analysis in the next step, we will only define two clusters.

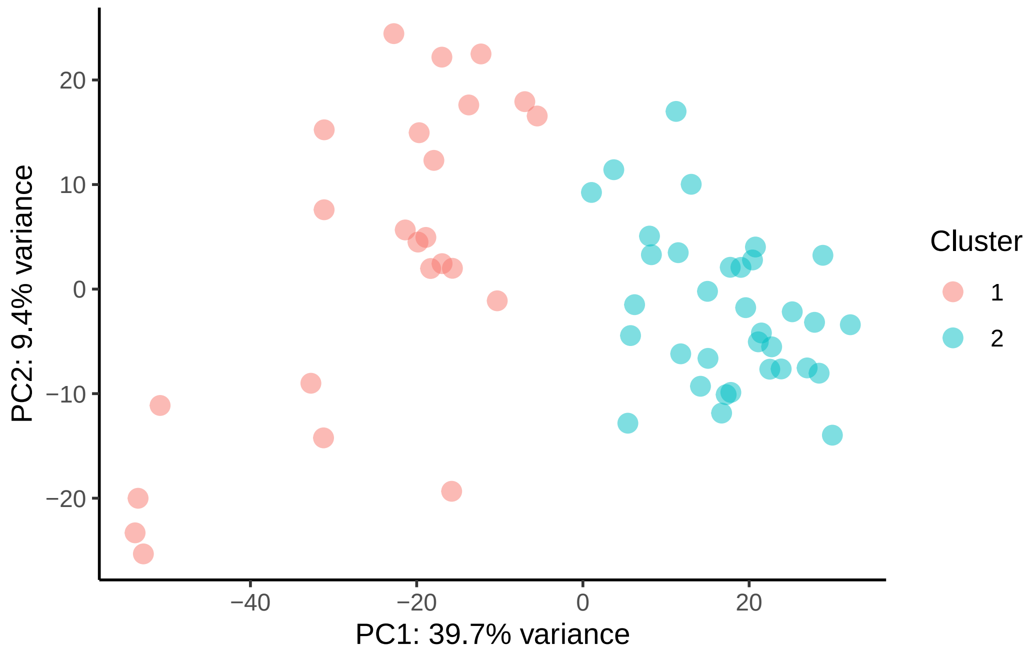
4.2 Dividing samples into two clusters

```
# Cutting the dendrogram above into 2 cluster
hc <- hclust(dist(t(high_var_mat)), method = "complete")
clusters <- cutree(hc, k = 2)

# Copy metadata for convenience, then keep only the columns you want to
display above the heatmap
sample_annotation_df <- ccle_meta_li
sample_annotation_df <- sample_annotation_df %>% dplyr::select(Pathology,
Gender, Race)

# Define a color map for each annotation column (values → colors)
sample_annotation_colors <- list(
  Pathology = c("primary" = "orange", "metastasis" = "green"),
  Gender = c("male" = "blue", "female" = "pink", "missing-info" = "grey"),
  Race = c("caucasian" = "yellow", "american_indian" = "red", "asian" =
  "brown", "missing-info" = "grey")
)

# Build a ComplexHeatmap annotation object using the data frame and color
maps
# 'annotation_label' sets the displayed labels for the annotation tracks
column_annotation <- HeatmapAnnotation(
  df = sample_annotation_df,
  col = sample_annotation_colors,
  annotation_label = c("Pathology", "Gender", "Race")
)

# Draw the heatmap of the top variable genes with hierarchical clustering
# 'top_annotation' adds the colored bars built above to the top of the
heatmap
Heatmap(high_var_mat,
        clustering_distance_columns = "euclidean",
        clustering_method_columns = "complete",
        clustering_distance_rows = "euclidean",
        clustering_method_rows = "complete",
        show_row_names = FALSE,
        show_column_names = FALSE,
        name = "VST_Counts",
        column_split = clusters,
        top_annotation = column_annotation)
```



Creating corresponding PCA plot:

```r
#Creating a PCA dataframe PC scores of PC1 and PC2 for plotting
pca_df <- data.frame(pca_res$x[, 1:2])

# Adding cluster assignments to PCA dataframe
pca_df$Cluster <- factor(clusters)

# Creating a scatterplot of PC1 vs. PC2 where each point is colored by
'Cluster'
ggplot(pca_df, aes(x = PC1, y = PC2, color = Cluster)) +
  geom_point(size = 3, alpha = 0.5) +
  scale_color_manual(values = hue_pal()(length(unique(clusters)))) +
  theme_classic() +
  labs(x = paste0("PC1: ", round(explained_variance[1], 1), "% variance"),
       y = paste0("PC2: ", round(explained_variance[2], 1), "% variance"),
       color = "Cluster")
```



5 Differential Expression Analysis

The core step of our workflow is Differential expression (DE) analysis which helps us move beyond just visualizing variation (like with PCA or clustering) and actually quantify which genes are changing between groups.

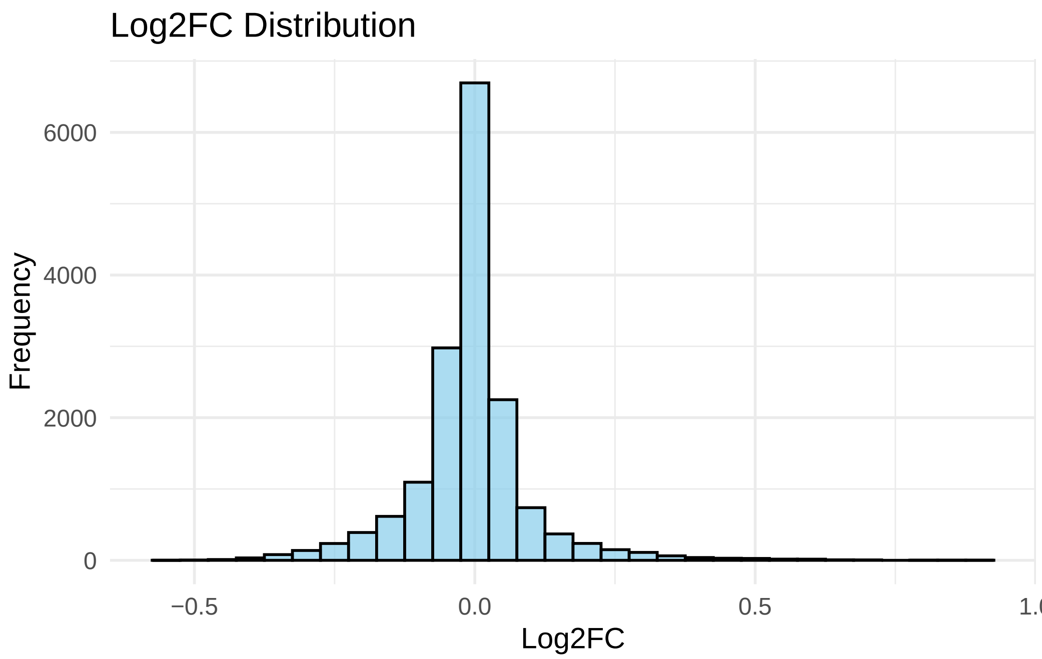
Before going ahead with final DE analysis using DESeq2, let's quickly explore clustering-based insights and find candidate genes.

5.1 log2FC method (using `vsd`)

This method offers quick exploratory comparison of average expression between clusters.

```
# Add cluster assignments as a new column in metadata
ccle_meta_li$Cluster <- factor(clusters)


# Subset expression matrix by sample cluster
cluster_1_data <- assay(vsd)[, ccle_meta_li$Cluster == 1]
cluster_2_data <- assay(vsd)[, ccle_meta_li$Cluster == 2]
```

```r
# Compute average expression per cluster (for each gene)
mean_cluster_1 <- rowMeans(cluster_1_data)
mean_cluster_2 <- rowMeans(cluster_2_data)


# Calculate log2 fold change in expression between clusters for each gene
log2FC <- log2((mean_cluster_2 + 1) / (mean_cluster_1 + 1))
# Note: Adding +1 avoids division by zero and stabilizes low counts
# +ve values: upregulated in Cluster 2
# -ve values: upregulated in Cluster 1


# Create histogram of log2FC values
ggplot(data.frame(log2FC = log2FC), aes(x = log2FC)) +
  geom_histogram(bins = 30, fill = "skyblue", color = "black", alpha = 0.7) +
  theme_minimal() +
  labs(x = "Log2FC", y = "Frequency", title = "Log2FC Distribution")
```



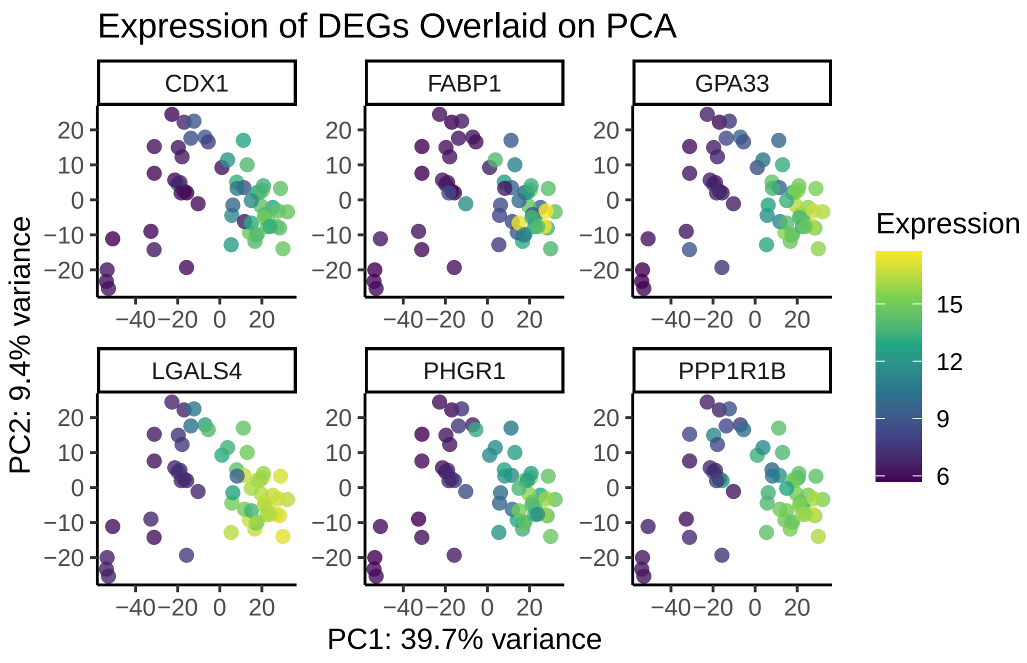
```r
# Create a DEG table with gene names, avg. expression across both clusters,
and log2 fold change
degs_df <- data.frame(
  gene = rownames(assay(vsd)),
  mean_expression = (mean_cluster_1 + mean_cluster_2) / 2,
  log2FC = log2FC
)

# Rank genes by absolute log2 FC
degs_ranking <- order(abs(degs_df$log2FC), decreasing = TRUE)

# Get 20 most differentially expressed genes between the two clusters
top_degs <- degs_df[degs_ranking, ][1:20, ]
print(top_degs)
```

```
             gene mean_expression    log2FC
LGALS4     LGALS4       11.659376 0.9232954
GPA33       GPA33       10.426875 0.8940838
PHGR1       PHGR1       10.018137 0.8729691
CDX1         CDX1        9.877519 0.8401491
PPP1R1B   PPP1R1B       11.005527 0.8127013
FABP1       FABP1        9.118397 0.7768574
CEACAM5   CEACAM5       11.258872 0.7213842
CDH17       CDH17       12.140444 0.7080249
ERN2         ERN2        9.785301 0.6997471
CFTR         CFTR       10.003429 0.6912633
ATP10B     ATP10B       10.055136 0.6871561
GPX2         GPX2       12.087213 0.6833856
DDC           DDC        9.904669 0.6735677
HMGCS2     HMGCS2        9.294870 0.6687608
HEPH         HEPH       10.909075 0.6660131
GUCY2C     GUCY2C        9.487327 0.6615789
C9orf152 C9orf152        9.853698 0.6601867
DPEP1       DPEP1       10.446434 0.6543002
REG4         REG4        9.042576 0.6260080
ANKS4B     ANKS4B        9.223101 0.6243931
```

Interpretation: This histogram peaks at 0, indicating that most genes are not strongly differentially expressed between the two clusters. And positive log2FC for all these indicate that all are upregulated in cluster 2 (right side of PCA plot). It also aligns with more red in cluster 2 from our hierarchical clustering in the previous section. Also not, many of these explain variance along PC1 and PC2 as previously explored.

Let's plot the expression of top 6 candidate DEGs we got from this method in PCA plots to see if they actually are differentially expressed.

```r
deg_genes <- c("LGALS4", "GPA33", "PHGR1", "CDX1", "PPP1R1B", "FABP1")

expr_mat <- assay(vsd)[deg_genes, ] # rows = genes, columns = samples

expr_df <- as.data.frame(t(expr_mat)) # rows = samples, columns = genes

expr_df$Sample <- rownames(expr_df)

pca_df$Sample <- rownames(pca_df)
plot_df <- merge(pca_df, expr_df, by = "Sample")

plot_long <- pivot_longer(
  plot_df,
  cols = all_of(deg_genes),
  names_to = "Gene",
  values_to = "Expression"
)

ggplot(plot_long, aes(x = PC1, y = PC2, color = Expression)) +
  geom_point(size = 2, alpha = 0.8) +
  facet_wrap(~ Gene, scales = "free", ncol = 3) +
  scale_color_viridis_c() +
  theme_classic() +
  labs(title = "Expression of DEGs Overlaid on PCA",
       x = paste0("PC1: ", round(explained_variance[1], 1), "% variance"),
       y = paste0("PC2: ", round(explained_variance[2], 1), "% variance"),
       color = "Expression")
```



They indeed look upregulated in cluster 2.

5.2 DESeq2 method

This method is designed for proper differential expression testing:

- **Statistical model:** It uses a negative binomial model for raw counts, estimating dispersion and library size.
- **Hypothesis testing:** Provides p-values and adjusted p-values (padj) controlling for multiple testing.
- **Handles replicates & covariates:** Can include batch effects or other variables in the design formula.
- **Variance shrinkage:** LFC shrinkage for more reliable effect size estimation (avoids inflated fold changes for low counts).

This means **DESeq2 identifies genes whose differences are unlikely due to chance**, whereas our quick approach before just identifies genes with the largest fold changes (which could also be noise).

```r
# Construct a DESeqDataSet object using raw counts and metadata with cluster
as design
dds <- DESeqDataSetFromMatrix(countData = data.frame(ccle_counts_li),
                              colData = ccle_meta_li,
                              design = ~ Cluster)
```

```
converting counts to integer mode
```

```r
# Relevel the factor so that Cluster 2 becomes the reference level, i.e.,
log2FC = log2(cluster1/cluster2)
dds$Cluster <- relevel(dds$Cluster, ref = "2")
# +ve : upregulated in cluster 1
# -ve : upregulated in cluster 2

# Filter out genes with counts < 100
dds <- dds[rowSums(counts(dds)) >= 100,]

# Run DESeq2 for differential expression analysis
dds <- DESeq(dds)
```

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates

mean-dispersion relationship

final dispersion estimates

fitting model and testing

-- replacing outliers and refitting for 2278 genes
-- DESeq argument 'minReplicatesForReplace' = 7
-- original counts are preserved in counts(dds)

estimating dispersions
```

```
fitting model and testing
```

Now, let's extract the DE results table from our `dds` object that was created using the design formula `~ Cluster` . By default, it compares the non-reference level (1 in our case) of `Cluster` against the reference level we set (`Cluster 2`in our case).

```r
res <- results(dds)
print(res)
```

```
log2 fold change (MLE): Cluster 1 vs 2
Wald test p-value: Cluster 1 vs 2
DataFrame with 16342 rows and 6 columns
            baseMean log2FoldChange     lfcSE      stat      pvalue        padj
           <numeric>      <numeric> <numeric> <numeric>   <numeric>   <numeric>
OR4F5        6.72162     -0.6069679  0.426763 -1.422260 1.54951e-01 2.79555e-01
OR4F29       5.52118     -0.4959717  0.405993 -1.221625 2.21849e-01 3.63637e-01
OR4F16       4.50862     -0.2513601  0.329146 -0.763673 4.45062e-01 5.99456e-01
SAMD11     324.99491      3.8358047  0.489459  7.836825 4.62081e-15 2.15752e-13
NOC2L    18224.72587     -0.0144394  0.121886 -0.118466 9.05698e-01 9.47559e-01
...              ...            ...       ...       ...         ...         ...
TMEM236      32.2841      -0.310022  0.460066 -0.673865   0.5003974    0.649420
VAMP7      6268.2880       0.306765  0.177657  1.726728   0.0842165    0.175321
WASH6P      219.4621       0.140358  0.216921  0.647048   0.5176011    0.663943
ZBED1      5458.3986      -0.111151  0.204367 -0.543882   0.5865226    0.721827
ZNF33B     3707.4020      -0.247956  0.206873 -1.198594   0.2306857    0.374512
```

Summary of DE analysis:

```r
summary(res)
```

```
out of 16342 with nonzero total read count
adjusted p-value < 0.1
LFC > 0 (up)       : 4450, 27%
LFC < 0 (down)     : 2253, 14%
outliers [1]       : 0, 0%
low counts [2]     : 0, 0%
(mean count < 0)
[1] see 'cooksCutoff' argument of ?results
[2] see 'independentFiltering' argument of ?results
```

Interpretation: Out of 16,342 expressed genes, **6,703 (41%) are significantly differentially expressed** at FDR < 0.1, with ~**27% up in Cluster 1** and ~**14% up in Cluster 2**, and no genes filtered out as low count or outliers.

We will now filter these results to get most striking DEGs.

```
# Step 1: Filter for padj < 0.001
res_sig <- res[which(res$padj < 0.001), ]

# Step 2: Reorder by absolute log2FoldChange (largest to smallest)
res_reordered <- res_sig[order(abs(res_sig$log2FoldChange), decreasing =
TRUE), ]
res_reordered <- as.data.frame(res_reordered)

# Step 3: Print the top genes
print(head(res_reordered))
```

```
         baseMean log2FoldChange     lfcSE      stat       pvalue         padj
SGIP1     584.827       8.693087 0.7345953  11.83385 2.609084e-32 3.279819e-29
PDPN     1308.174       8.671718 0.7802106  11.11459 1.065431e-28 4.974648e-26
COL3A1  40355.465       8.477241 0.7414220  11.43376 2.835637e-30 1.853599e-27
SLC26A3  1066.221      -8.372524 0.7910717 -10.58377 3.543910e-26 9.341060e-24
OLFM4    3753.691      -8.196661 0.8014044 -10.22787 1.487535e-24 2.964549e-22
GPA33   17000.981      -8.170708 0.5054662 -16.16470 8.947815e-59 1.462252e-54
```

Identifying top 3 DEGs in both the clusters, respectively:

```
# Top 3 upregulated in Cluster 1
top3_cluster1 <- rownames(res_reordered[res_reordered$log2FoldChange > 0,
])[1:3]
print(top3_cluster1)
```

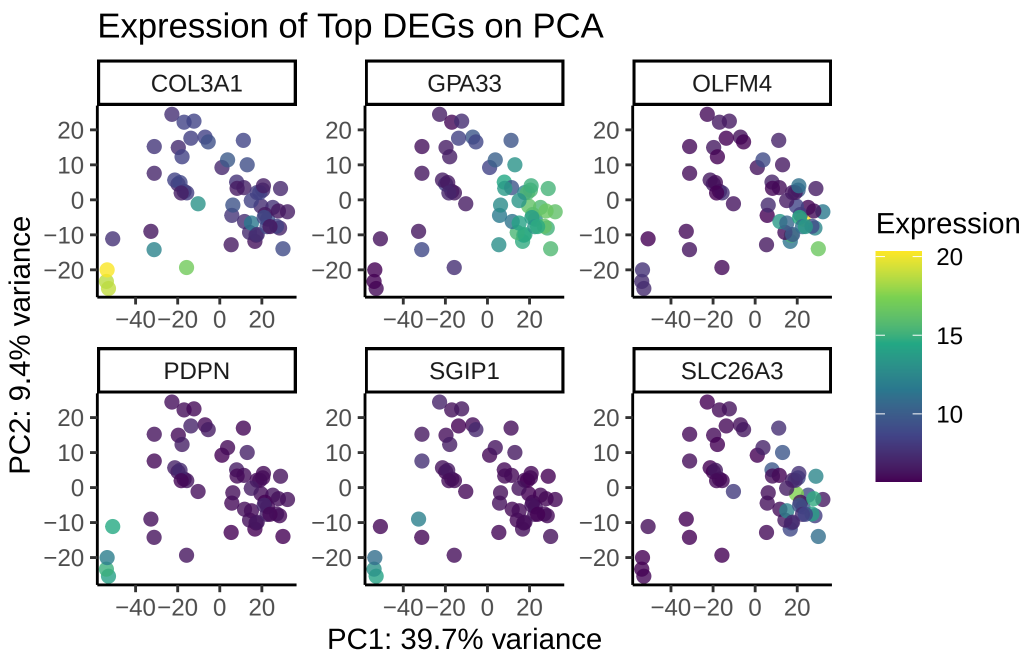
```
[1] "SGIP1"  "PDPN"   "COL3A1"
```

```
# Top 3 upregulated in Cluster 2
top3_cluster2 <- rownames(res_reordered[res_reordered$log2FoldChange < 0,
])[1:3]
print(top3_cluster2)
```

```
[1] "SLC26A3" "OLFM4"   "GPA33"
```

PCA visualization of these:

```r
top6_genes <- c(top3_cluster1, top3_cluster2)

expr_mat <- assay(vsd)[top6_genes, ]
expr_df <- as.data.frame(t(expr_mat))
expr_df$Sample <- rownames(expr_df)

# Merge with PCA info
pca_df$Sample <- rownames(pca_df)
plot_df <- merge(pca_df, expr_df, by = "Sample")

# Reshape
plot_long <- pivot_longer(plot_df, cols = all_of(top6_genes), names_to =
"Gene", values_to = "Expression")

# Plot
ggplot(plot_long, aes(x = PC1, y = PC2, color = Expression)) +
  geom_point(size = 2, alpha = 0.8) +
  facet_wrap(~ Gene, scales = "free", ncol = 3) +
  scale_color_viridis_c() +
  theme_classic() +
  labs(title = "Expression of Top DEGs on PCA",
       x = paste0("PC1: ", round(explained_variance[1], 1), "% variance"),
       y = paste0("PC2: ", round(explained_variance[2], 1), "% variance"),
       color = "Expression")
```



Based on this plot, there is some differential expression pattern in different clusters for each of these genes, so I am convinced these genes are DEGs.

- **SGIP1, PDPN, COL3A1 appear upregulated in cluster 1** (yellow-green spots on left)
- **SLC26A3, OLFM4, GPA33 appear upregulated in cluster 2** (yellow-green spots on right)

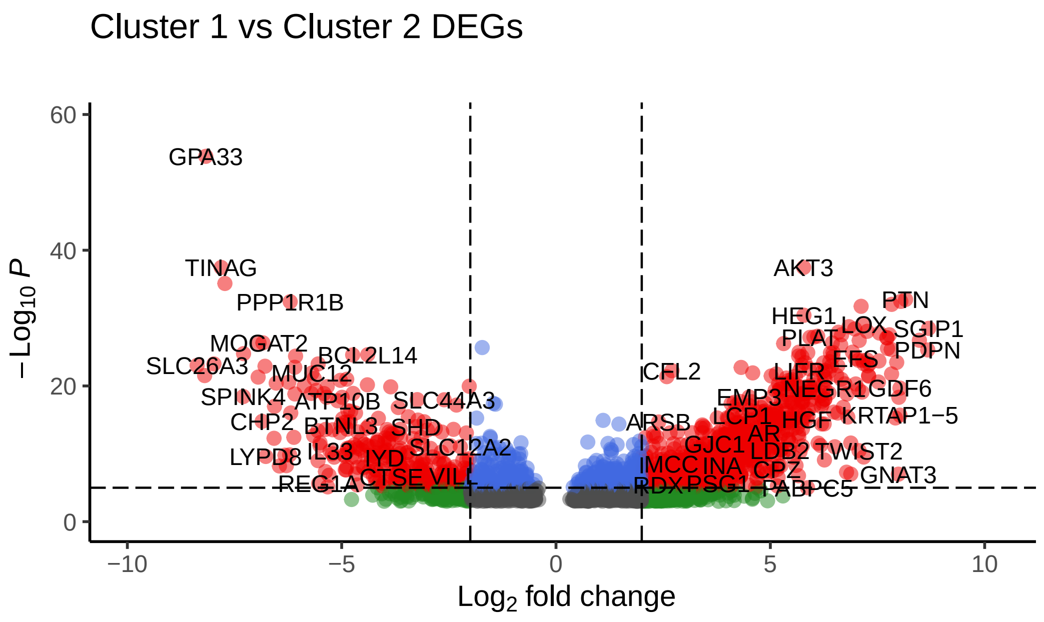
But DEGs in cluster 1 could have fallen in clusters 3/4 if we defined 4 clusters.

GPA33 is most strikingly differentially expressed and was also found during previous data explorations.

5.3 Volcano Plot

A volcano plot is a very common next step after DESeq2 results because it quickly visualizes which genes have the largest and most statistically significant differences between groups, combining both magnitude and significance of change in a single intuitive figure.

```
EnhancedVolcano(
  res_reordered,                   #Input dataframe of DE results
  x = "log2FoldChange",            #Set the x-axis to use the log2FoldChange
  y = "padj",                      #Set the y-axis to use the adjusted p-value
  (padj)
  lab = rownames(res_reordered),  #Label each point (gene) using the row
  names of res_reordered
  pCutoff = 0.00001,              #Set a stringent significance threshold for
  adjusted p-value (padj)
  FCcutoff = 2,                   #Set the threshold for log2 fold change
  magnitude, >2 will be considered biologically significant.
  labSize = 3,                    #Set the font size for the gene labels
  title = "Cluster 1 vs Cluster 2 DEGs",
  subtitle = "",
  caption = ""
) +
  theme_classic() +
  theme(legend.position = "none")
```



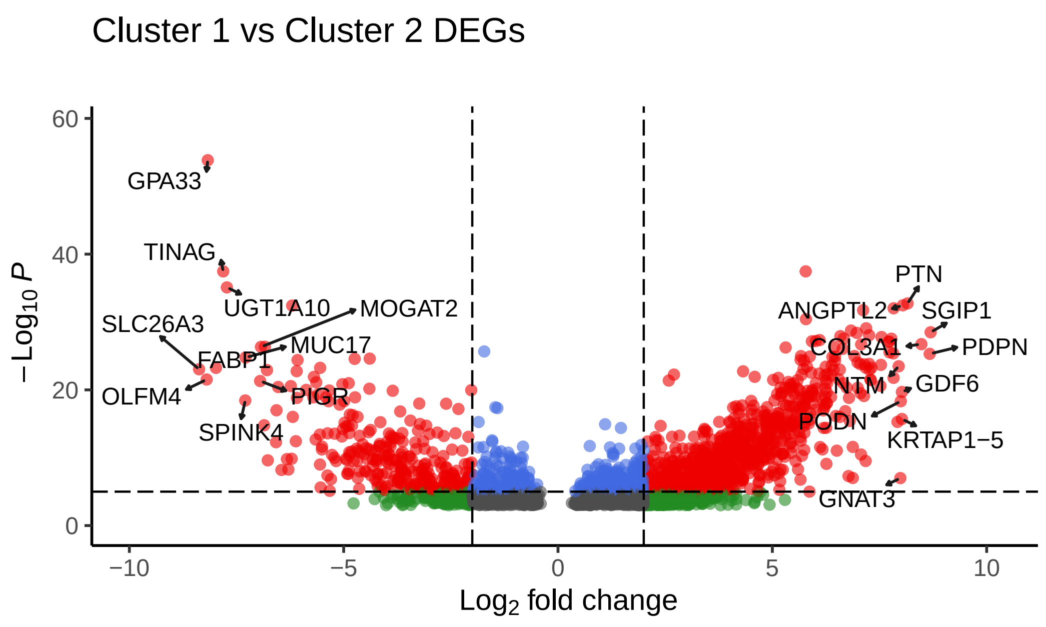
For a more readable volcano plot, let's focus on top 10 DEGs on each side.

```
# Step 1: Identify top 10 DEGs on each side
top_up <- res_reordered[res_reordered$log2FoldChange > 0, ]
top_up <- head(top_up[order(-top_up$log2FoldChange), ], 10)

top_down <- res_reordered[res_reordered$log2FoldChange < 0, ]
top_down <- head(top_down[order(top_down$log2FoldChange), ], 10)
```

```r
top_genes <- c(rownames(top_up), rownames(top_down))
print(top_genes)
```

```
 [1] "SGIP1"    "PDPN"     "COL3A1"   "PTN"      "ANGPTL2"  "GDF6"
 [7] "KRTAP1-5" "PODN"     "GNAT3"    "NTM"      "SLC26A3"  "OLFM4"
[13] "GPA33"    "FABP1"    "TINAG"    "UGT1A10"  "SPINK4"   "MUC17"
[19] "PIGR"     "MOGAT2"
```

```r
# Step 2: Volcano plot with smaller, more transparent points and selective
labeling
EnhancedVolcano(
  res_reordered,
  x = "log2FoldChange",
  y = "padj",
  lab = rownames(res_reordered),
  selectLab = top_genes,
  pCutoff = 0.00001,
  FCcutoff = 2,
  pointSize = 1.5,        # smaller dots
  labSize = 3,
  colAlpha = 0.6,         # more transparent dots
  title = "Cluster 1 vs Cluster 2 DEGs",
  subtitle = "",
  caption = "",
  drawConnectors = TRUE,
  widthConnectors = 0.5,
  max.overlaps = Inf
) +
  theme_classic() +
  theme(legend.position = "none")
```



Interpretation: The genes on right (+ve log2FC values) are upregulated in cluster 1 and vice versa. Also, Y-axis (−log10 p-value) represents the significance of differential expression with higher values = more significant so as seen before, **GPA33** has strikingly differential expression and is also very significant.

6 Interpretation

Overall, we defined **two groups** of large intestine cancer cell lines. The metadata couldn't help with defining variance sources so we looked at top DEGs in each cluster and based on that, following is the interpretation of biology of each group.

6.1 Normal Biological Functions of top 10 DEGs in each cluster:

6.1.1 Top 10 DEGs in Cluster 1 (High in Cluster 1):

SGIP1: Endocytic protein involved in membrane trafficking and receptor internalization
PDPN: Podoplanin - lymphatic vessel marker and stromal cell surface glycoprotein
COL3A1: Type III collagen - extracellular matrix protein providing structural integrity
PTN: Pleiotrophin - growth factor involved in cell migration and angiogenesis
ANGPTL2: Angiopoietin-like protein 2 - regulates angiogenesis and stem cell maintenance
GDF6: Growth differentiation factor 6 - TGF- family member for bone/joint formation

KRTAP1-5: Keratin-associated protein - structural protein in hair/epithelial cells
PODN: Podocan - extracellular matrix proteoglycan
GNAT3: G-protein alpha subunit - taste transduction signaling
NTM: Neurotrimin - cell adhesion molecule in neural development

6.1.2 Top 10 DEGs in Cluster 2 (High in Cluster 2):

SLC26A3: Chloride/bicarbonate exchanger - essential for intestinal ion transport
OLFM4: Olfactomedin 4 - intestinal stem cell marker and antimicrobial protein
GPA33: Glycoprotein A33 - intestinal epithelial cell surface antigen (we discussed earlier)
FABP1: Fatty acid binding protein 1 - lipid metabolism in liver and intestine
TINAG: Tubulointerstitial nephritis antigen - extracellular matrix protein
UGT1A10: UDP-glucuronosyltransferase - drug/xenobiotic detoxification enzyme
SPINK4: Serine protease inhibitor - antimicrobial defense in intestine
MUC17: Mucin 17 - membrane-bound mucin protecting intestinal epithelial barrier
PIGR: Polymeric immunoglobulin receptor - transports antibodies across epithelium
MOGAT2: Monoacylglycerol acyltransferase 2 - lipid metabolism enzyme

6.2 Role in Large Intestine Cancer:

6.2.1 Cluster 1 (Stromal/Mesenchymal-like):

- **PDPN**: Cancer-associated fibroblast marker; promotes invasion and metastasis
- **COL3A1**: Overexpressed in CRC epithelium; promotes proliferation via PI3K/AKT signaling
- **PTN, ANGPTL2, GDF6**: Growth factors supporting angiogenesis and tumor progression
- **Overall phenotype**: Enhanced stromal activation, EMT, invasion, and poor prognosis

6.2.2 Cluster 2 (Epithelial/Differentiated-like):

- **SLC26A3, UGT1A10, PIGR**: Maintain differentiated intestinal epithelial functions (like transportation)
- **OLFM4**: Intestinal stem cell marker; indicates preserved stem cell hierarchy
- **MUC17**: Protective mucin barrier function
- **MOGAT2**: Tumor suppressor in CRC; loss promotes tumor progression
- **Overall phenotype**: Preserved intestinal epithelial identity and metabolic functions

6.3 Biological Difference Between Clusters:

Cluster 1 likely represents **"Stromal-Activated/Mesenchymal-like Cancer"**:

- High stromal gene expression (PDPN, COL3A1, growth factors)
- Enhanced cancer-associated fibroblast activity
- Increased EMT, invasion, and metastatic potential
- Corresponds to **CMS4-like** aggressive, stromal-infiltrated tumors

Cluster 2 likely represents **"Epithelial-Differentiated Cancer"**:

- Preserved intestinal epithelial gene expression (transporters, mucins, metabolic enzymes)
- Maintained differentiated cellular functions
- Better preservation of normal intestinal architecture
- Corresponds to **CMS2-like** canonical, differentiated tumors

Summary: The clusters likely distinguish between **"Stromal-Driven Aggressive"** vs **"Epithelial-Differentiated"** colorectal cancer phenotypes, representing fundamentally different mechanisms of tumorigenesis and clinical behavior. This difference in biology of clusters 1 & 2 was also somewhat captured in PCA in previous sections. Finally, we should define more clusters in the future to obtain finer details of biological differences between large intestine cancer cell lines in this CCLE dataset.