FINAL PROJECT

#1. install necessary libraries

```
library(Seurat)
```

```
## Loading required package: SeuratObject
```

```
## Loading required package: sp
```

```
##
## Attaching package: 'SeuratObject'
```

```
## The following objects are masked from 'package:base':
##
##     intersect, t
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

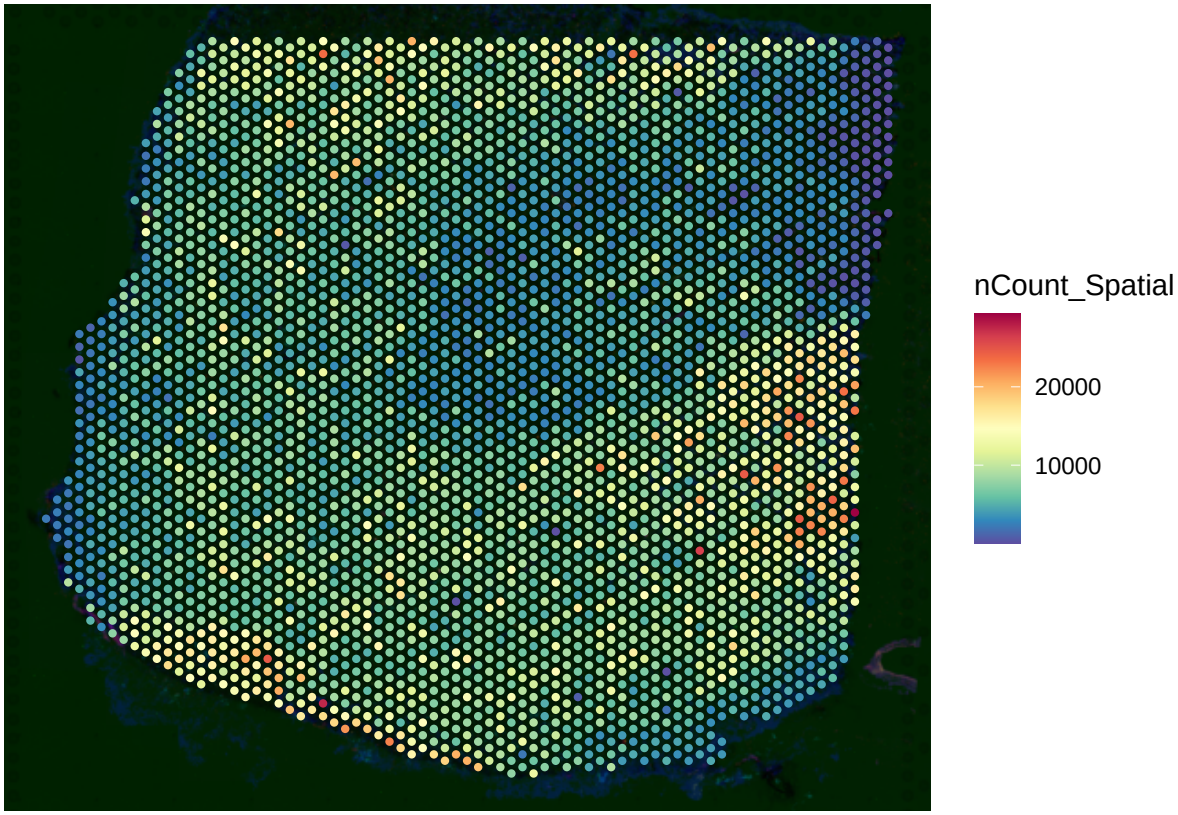
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(patchwork)

options(future.globals.maxSize = 3 * 1024^3)  # Set to 3 GiB
```

#2 Load Control 10x Data

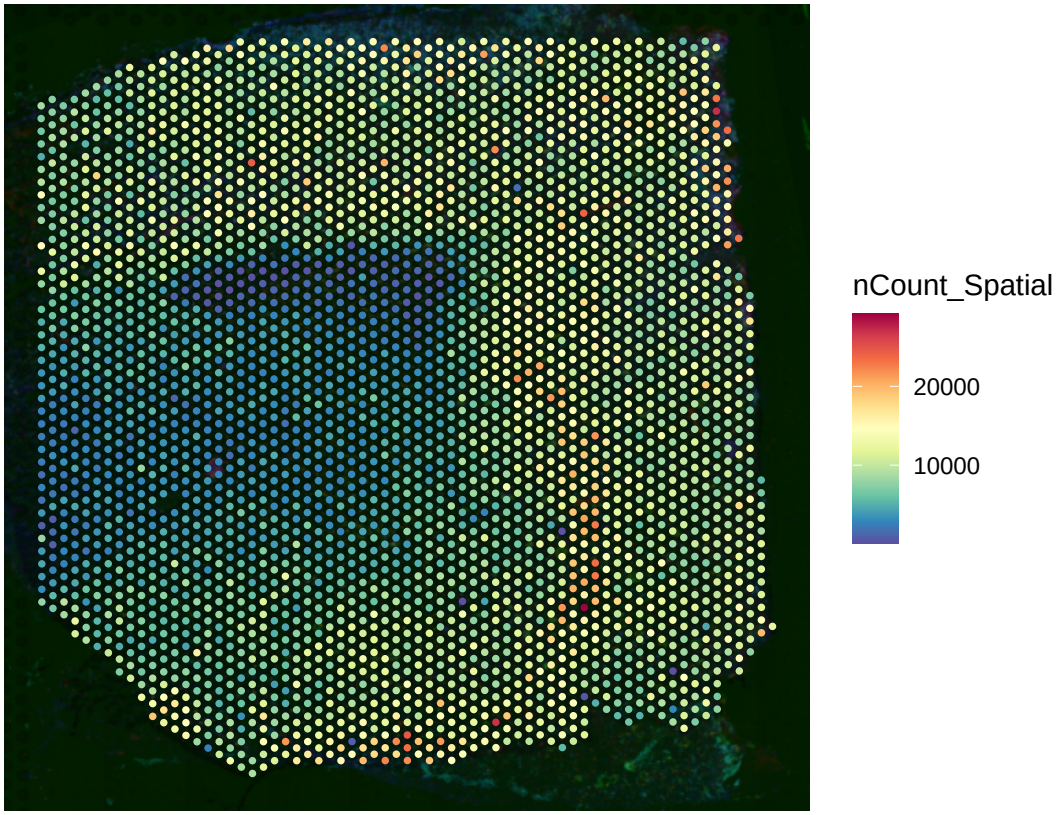
```
control = Load10X_Spatial(
     "/Users/cougerjaramillo/Desktop/MS/Intersession 2026/Applications/counts_and_images/1-1",
     filename = "filtered_feature_bc_matrix.h5",
     assay = "Spatial",
     slice = "slice1",
     bin.size = NULL,
     filter.matrix = TRUE,
     to.upper = FALSE,
     image = NULL
)
# Set orig.ident of control to 0
control@meta.data$orig.ident <- "0"
SpatialFeaturePlot(control, features = "nCount_Spatial") + theme(legend.position = "right")
```




```
control
```

```
## An object of class Seurat
## 36601 features across 3742 samples within 1 assay
## Active assay: Spatial (36601 features, 0 variable features)
##  1 layer present: counts
##  1 spatial field of view present: slice1
```

```
AD = Load10X_Spatial(
     "/Users/cougerjaramillo/Desktop/MS/Intersession 2026/Applications/counts_and_images/2-8",
     filename = "filtered_feature_bc_matrix.h5",
     assay = "Spatial",
     slice = "slice2",
     bin.size = NULL,
     filter.matrix = TRUE,
     to.upper = FALSE,
     image = NULL
 )
# Set orig.ident of AD to 1
AD@meta.data$orig.ident <- "1"
SpatialFeaturePlot(AD, features = "nCount_Spatial") + theme(legend.position = "right")
```



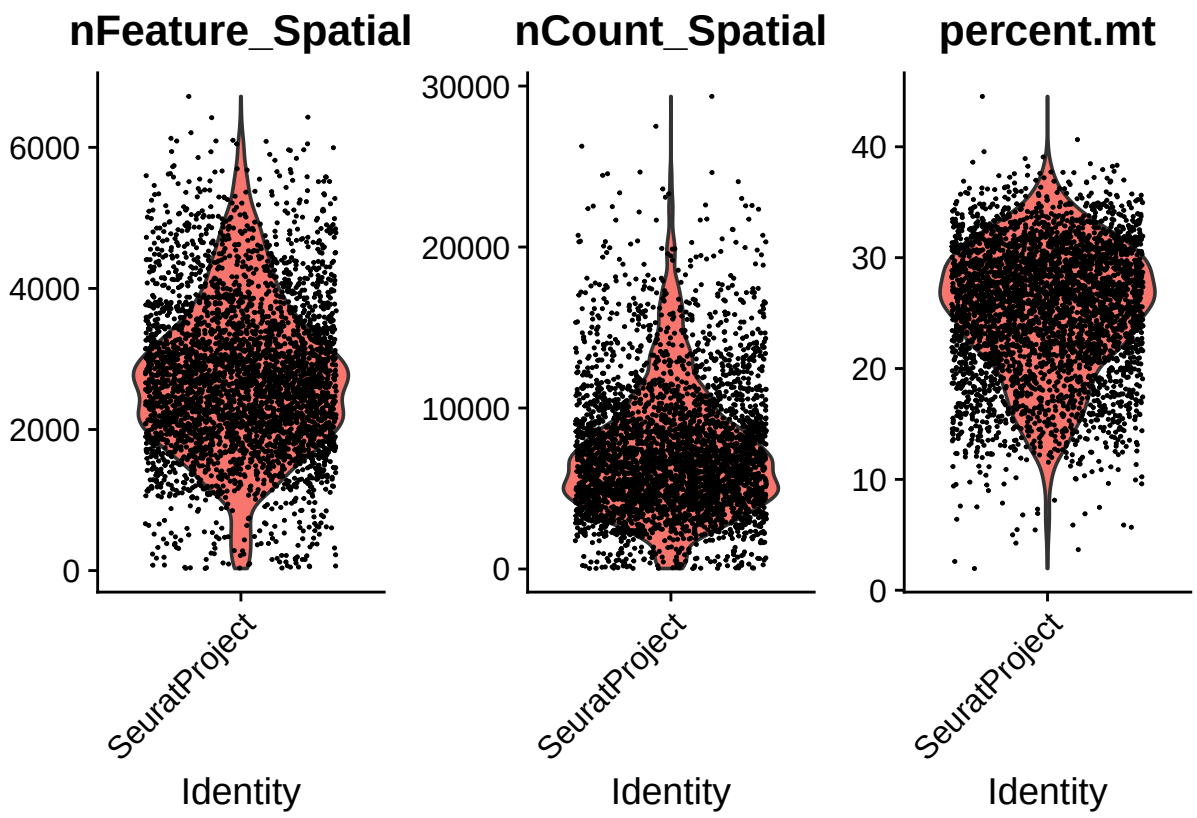

```
AD
```

```
## An object of class Seurat
## 36601 features across 3445 samples within 1 assay
## Active assay: Spatial (36601 features, 0 variable features)
##  1 layer present: counts
##  1 spatial field of view present: slice2
```

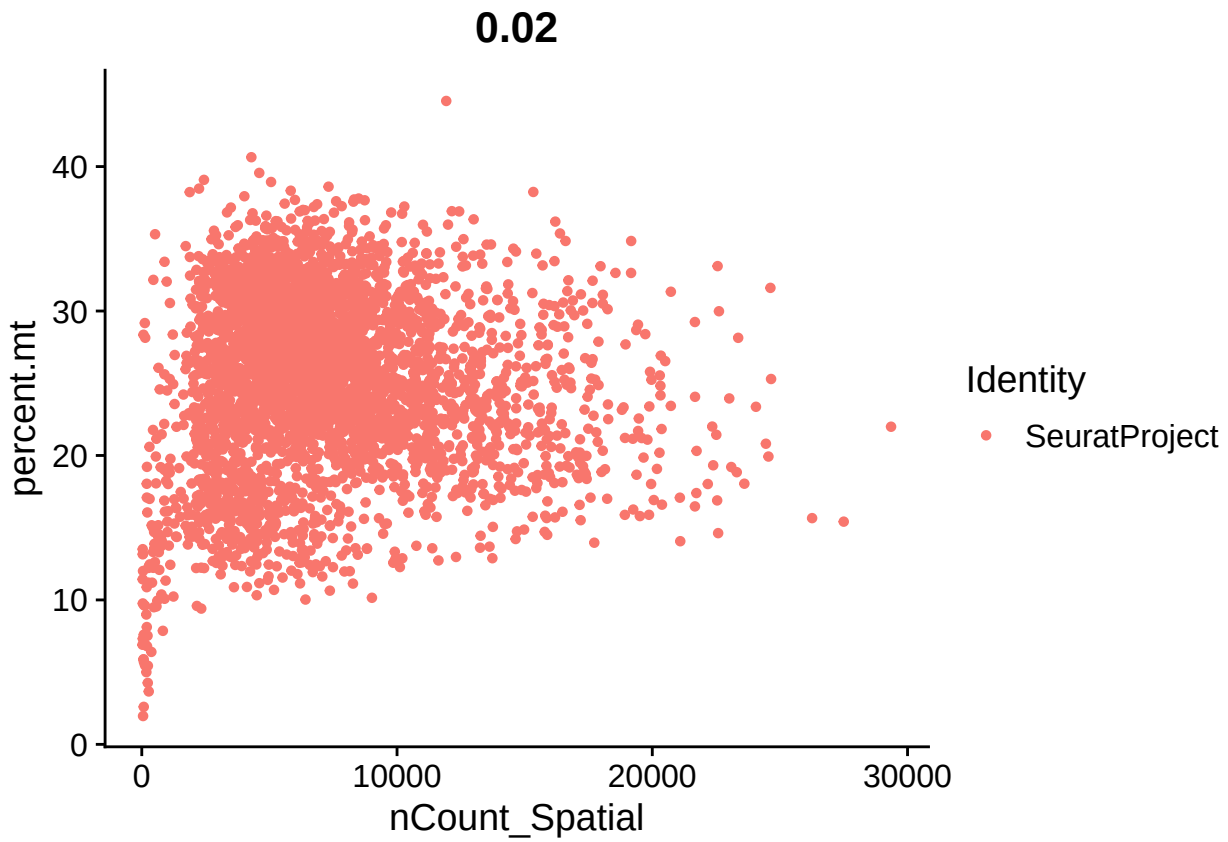
#3. Pre-processing

```
#control
#Mitochondrial DNA counts
control[["percent.mt"]] <- PercentageFeatureSet(control, pattern = "^MT-")#
#Visualize
VlnPlot(control, features = c("nFeature_Spatial", "nCount_Spatial", "percent.mt"), ncol = 3)
```

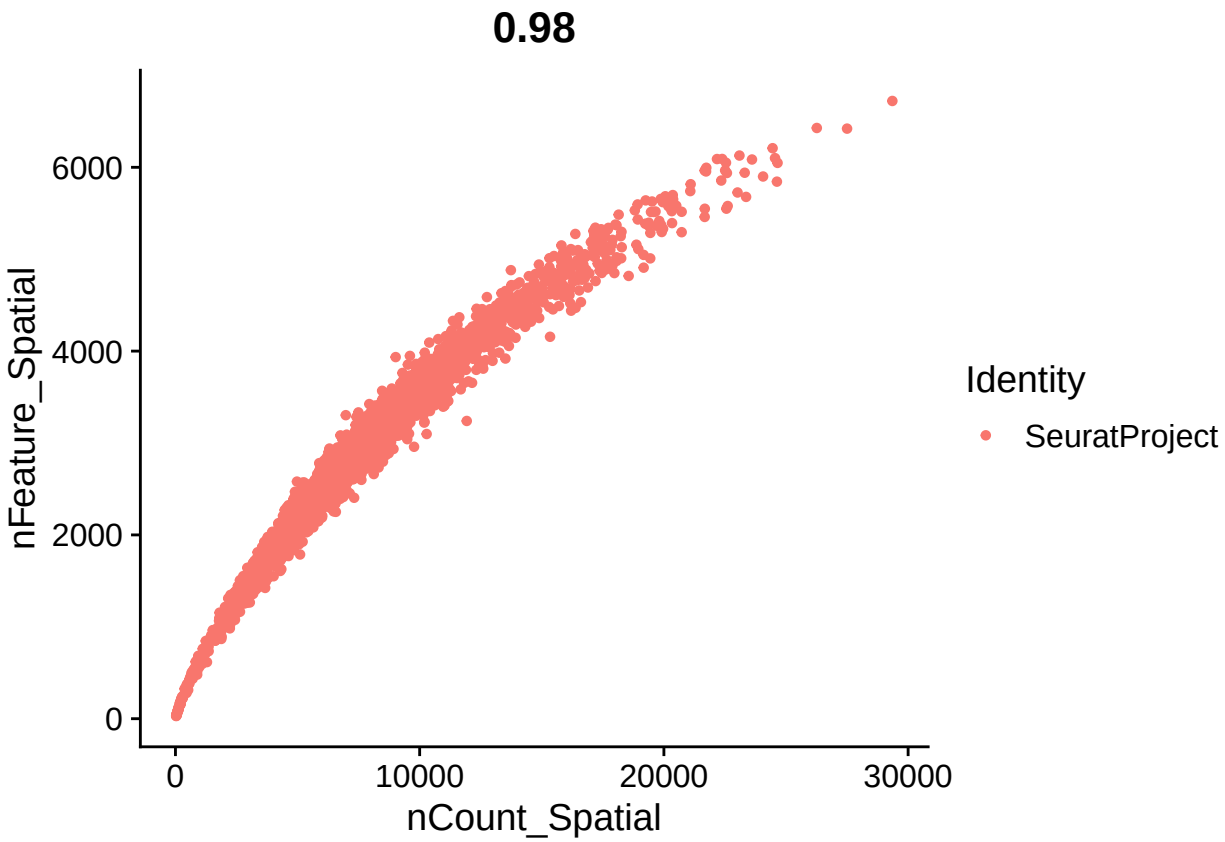
```
## Warning: Default search for "data" layer in "Spatial" assay yielded no results;
## utilizing "counts" layer instead.
```




```
plot1 <- FeatureScatter(control, feature1 = "nCount_Spatial", feature2 = "percent.mt")
plot2 <- FeatureScatter(control, feature1 = "nCount_Spatial", feature2 = "nFeature_Spatial")
plot1
```


0.02
percent.mt
nCount_Spatial
0
10
20
30
40
0
10000
20000
30000
Identity
SeuratProject

```
plot2
```




```
#Subset for high quality data
control <- subset(control, subset = nFeature_Spatial > 500 & nFeature_Spatial < 6000 & percent.mt < 40)
```

```
## Warning: Not validating Centroids objects
```

```
## Warning: Not validating Centroids objects
```

```
## Warning: Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
```

```
## Warning: Not validating Seurat objects
```

```
#Log normalize
control <- NormalizeData(control, normalization.method = "LogNormalize", scale.factor = 10000)
```

```
## Normalizing layer: counts
```

```
#Feature Selection
control <- FindVariableFeatures(control, selection.method = "vst", nfeatures = 2000)
```

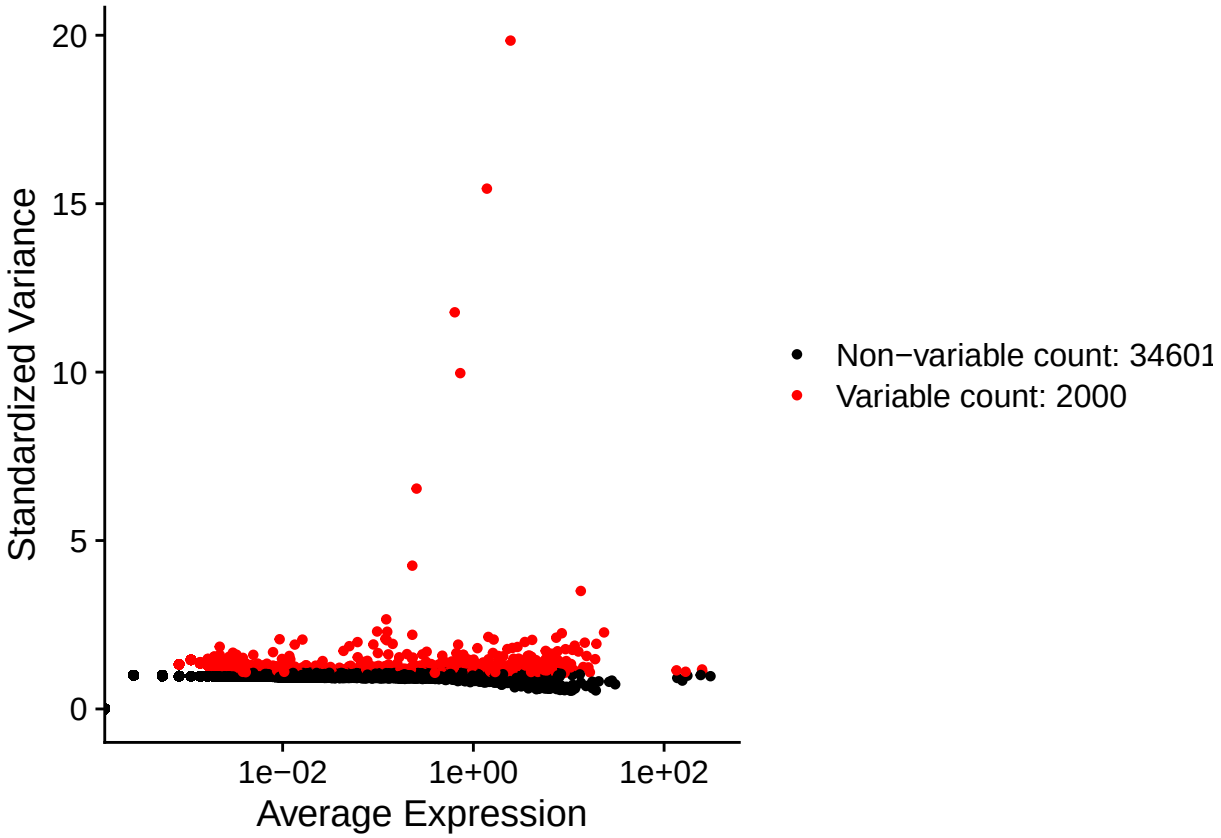
```
## Finding variable features for layer counts
```

```
# Identify the 10 most highly variable genes
top10 <- head(VariableFeatures(control), 10)
# plot variable features with and without labels
plot1 <- VariableFeaturePlot(control)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
```

```
## When using repel, set xnudge and ynudge to 0 for optimal results
```

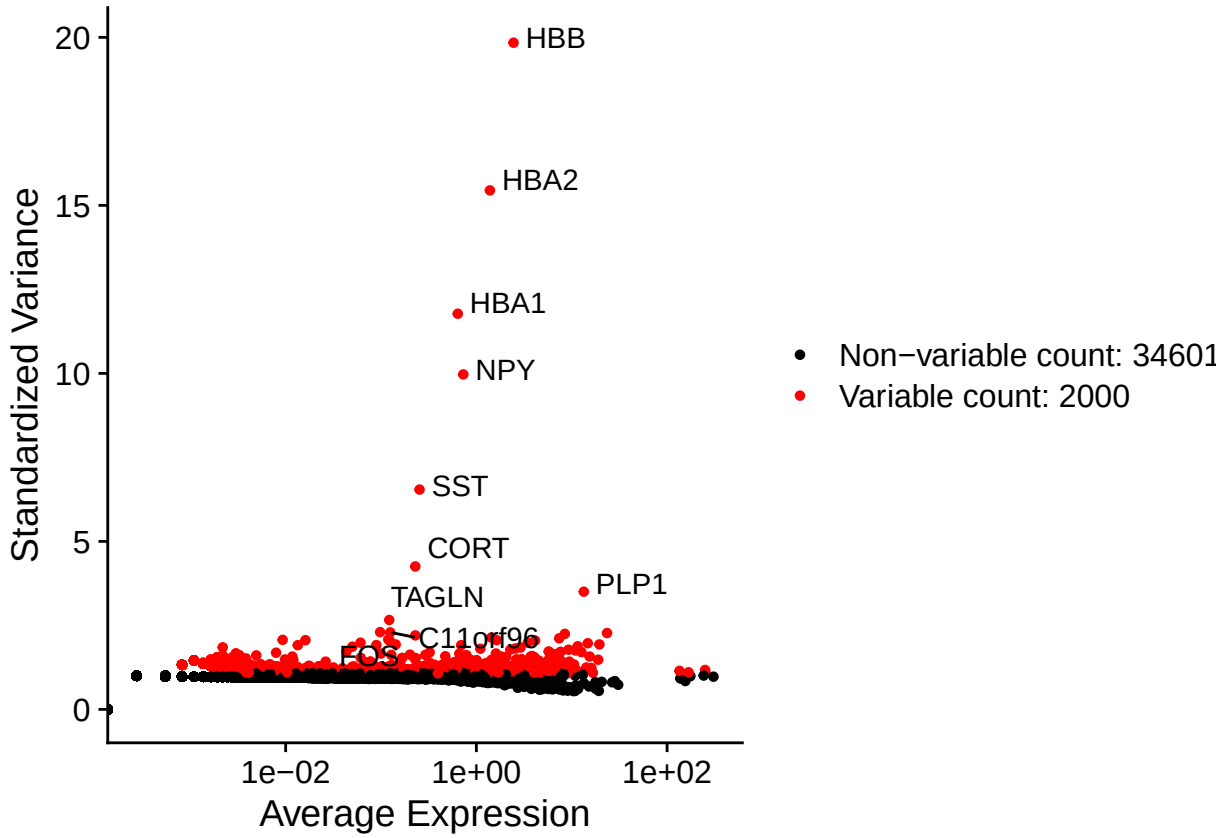
```
plot1
```

```
## Warning in scale_x_log10(): log-10 transformation introduced infinite values.
```




```
plot2
```

```
## Warning in scale_x_log10(): log-10 transformation introduced infinite values.
```




```
#Scale Data
all.genes <- rownames(control)
control <- ScaleData(control, features = all.genes)
```

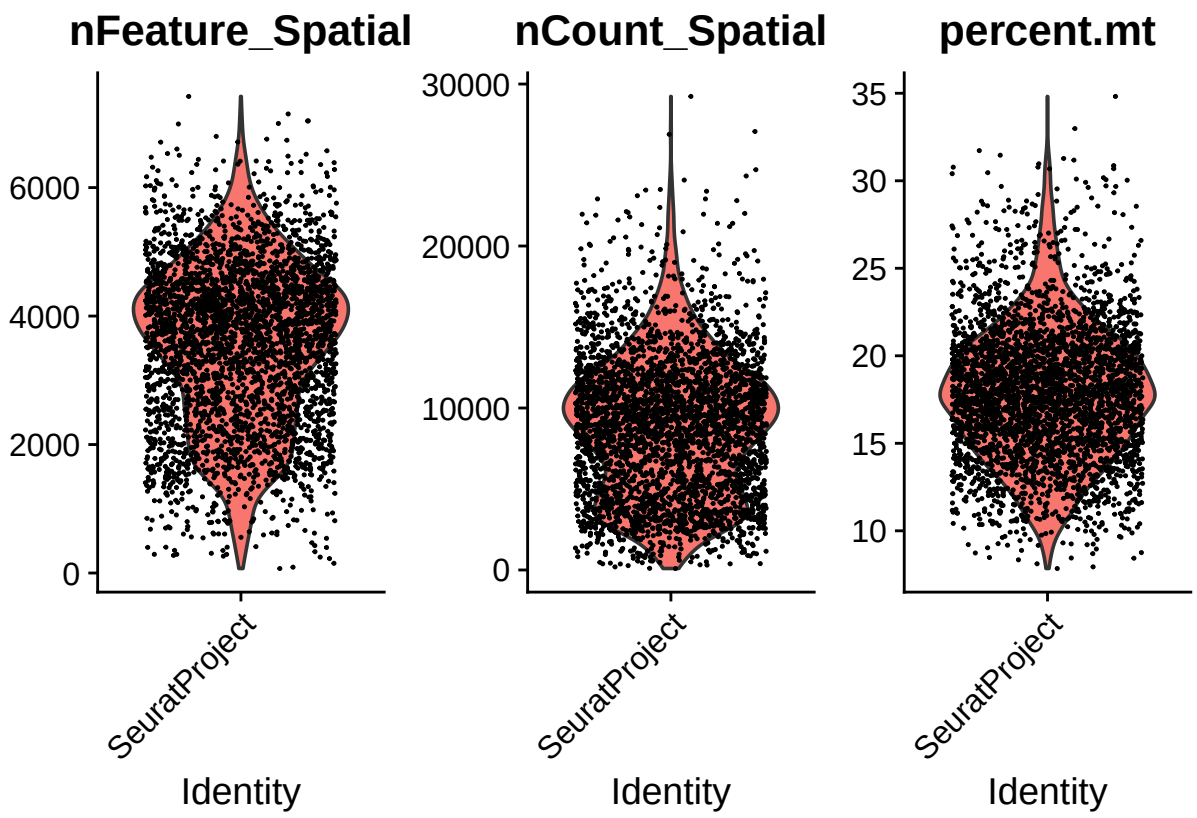
```
## Centering and scaling data matrix
```

```
#check final object for QC
control
```

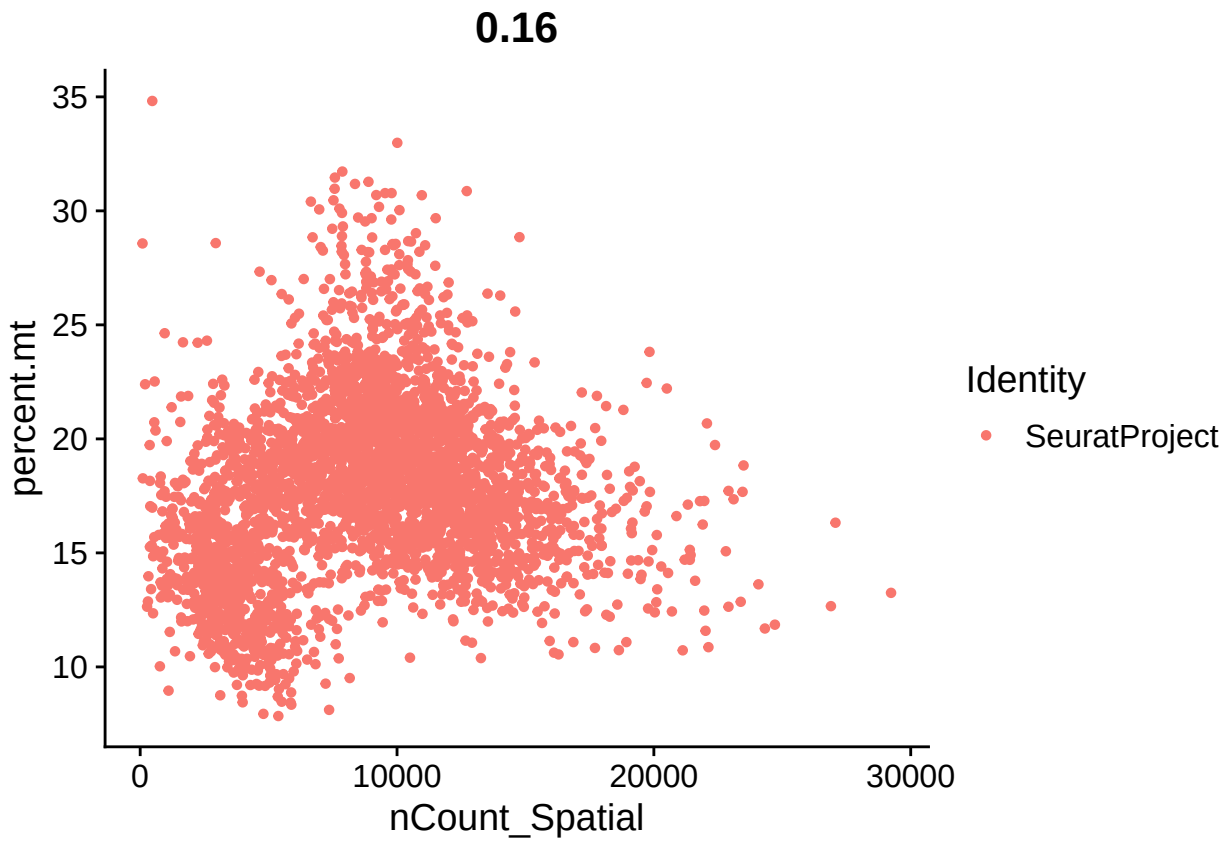
```
## An object of class Seurat
## 36601 features across 3657 samples within 1 assay
## Active assay: Spatial (36601 features, 2000 variable features)
##  3 layers present: counts, data, scale.data
##  1 spatial field of view present: slice1
```

```
#AD
#Mitochondrial DNA counts
AD[["percent.mt"]] <- PercentageFeatureSet(AD, pattern = "^MT-")#
#Visualize
VlnPlot(AD, features = c("nFeature_Spatial", "nCount_Spatial", "percent.mt"), ncol = 3)
```

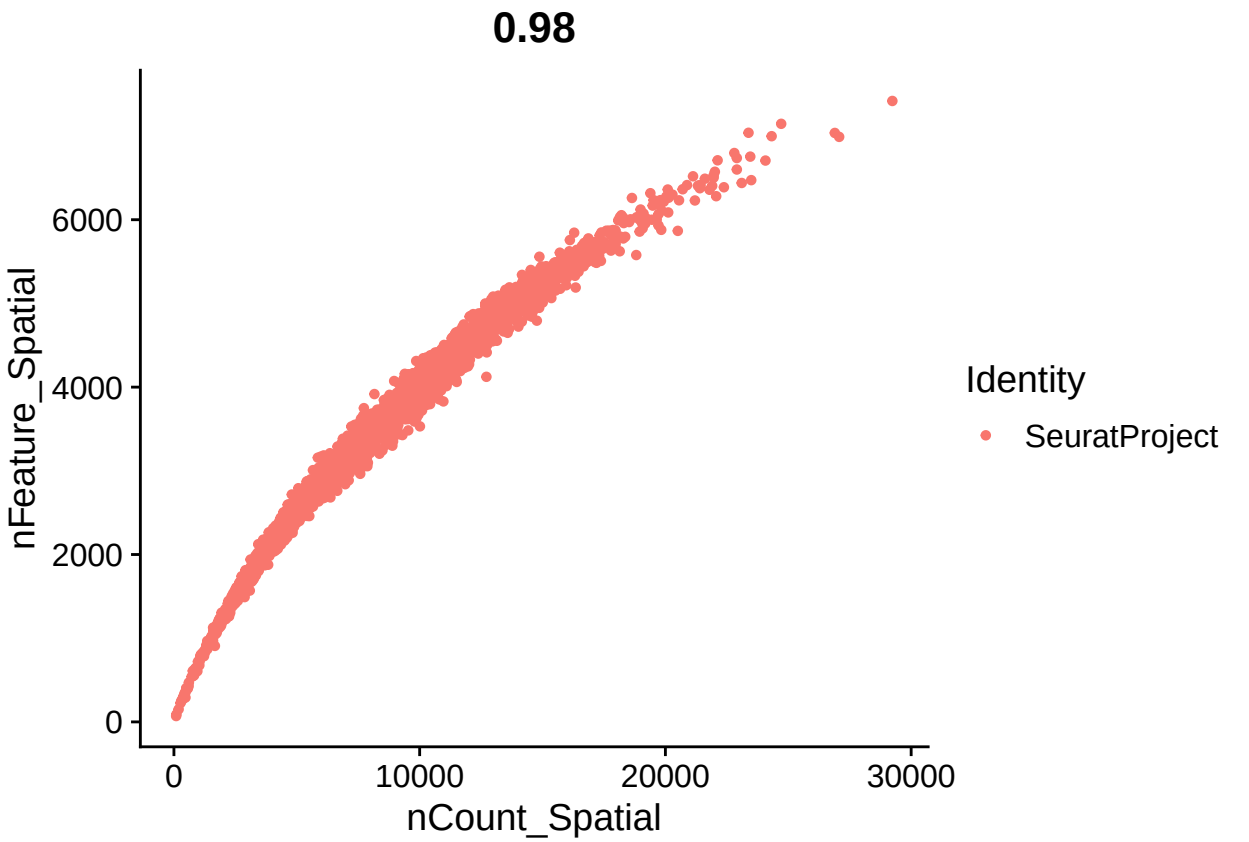
```
## Warning: Default search for "data" layer in "Spatial" assay yielded no results;
## utilizing "counts" layer instead.
```




```
plot1 <- FeatureScatter(AD, feature1 = "nCount_Spatial", feature2 = "percent.mt")
plot2 <- FeatureScatter(AD, feature1 = "nCount_Spatial", feature2 = "nFeature_Spatial")
plot1
```


0.16
35
30
25
20
15
10
percent.mt
0
10000
20000
30000
nCount_Spatial
Identity
SeuratProject

```
plot2
```




```
#Subset for high quality data
AD <- subset(AD, subset = nFeature_Spatial > 500 & nFeature_Spatial < 6000 & percent.mt < 40)
```

```
## Warning: Not validating Centroids objects
```

```
## Warning: Not validating Centroids objects
```

```
## Warning: Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
```

```
## Warning: Not validating Seurat objects
```

```
#Log normalize
AD <- NormalizeData(AD, normalization.method = "LogNormalize", scale.factor = 10000)
```

```
## Normalizing layer: counts
```

```
#Feature Selection
AD <- FindVariableFeatures(AD, selection.method = "vst", nfeatures = 2000)
```

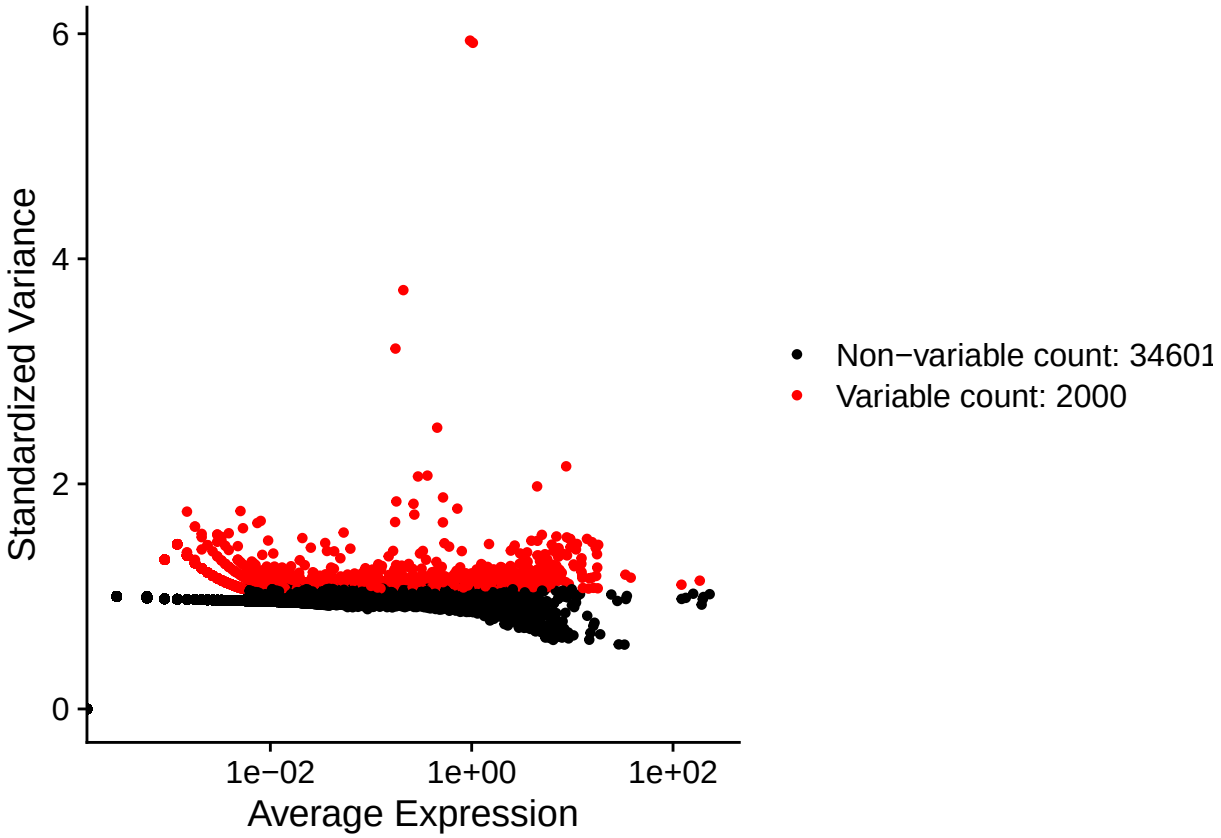
```
## Finding variable features for layer counts
```

```
# Identify the 10 most highly variable genes
top10 <- head(VariableFeatures(AD), 10)
# plot variable features with and without labels
plot1 <- VariableFeaturePlot(AD)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
```

```
## When using repel, set xnudge and ynudge to 0 for optimal results
```

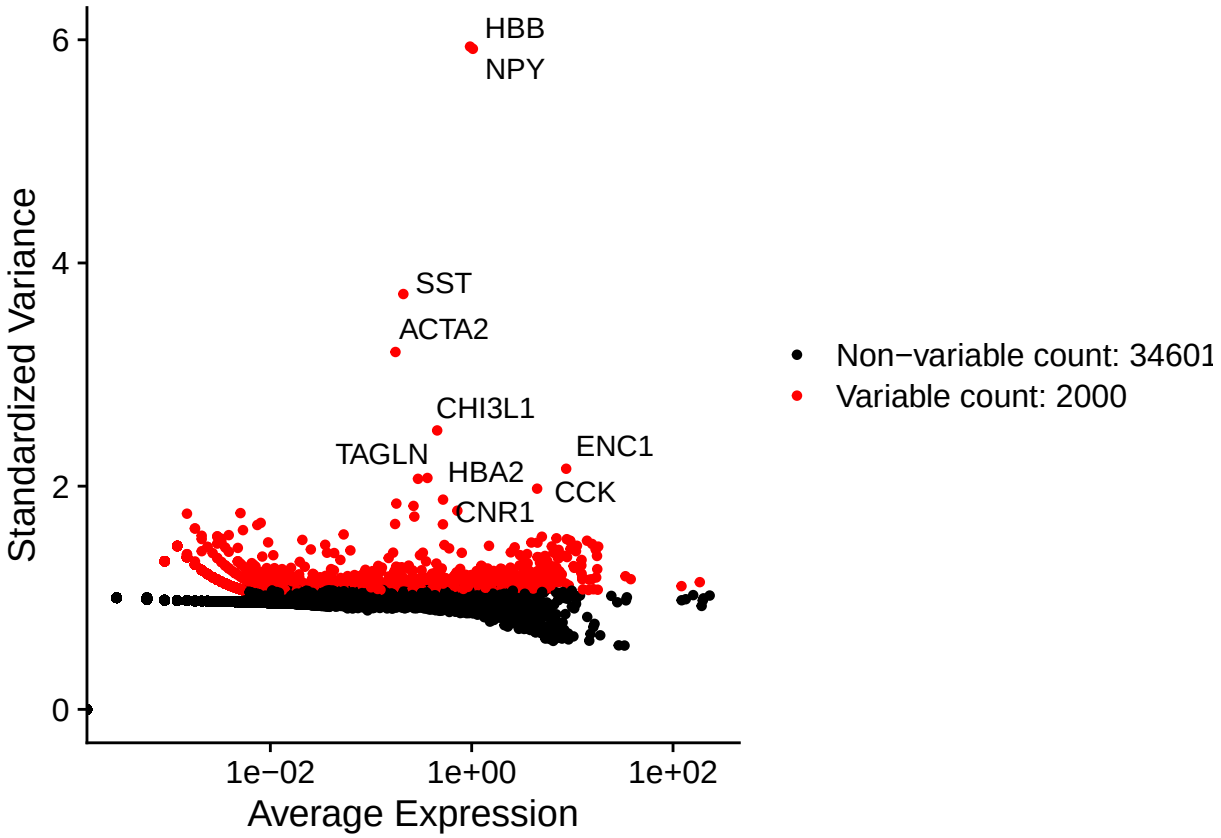
```
plot1
```

```
## Warning in scale_x_log10(): log-10 transformation introduced infinite values.
```




```
plot2
```

```
## Warning in scale_x_log10(): log-10 transformation introduced infinite values.
```




```
#Scale Data
all.genes <- rownames(AD)
AD <- ScaleData(AD, features = all.genes)
```

```
## Centering and scaling data matrix
```

```
#check final object for QC
AD
```

```
## An object of class Seurat
## 36601 features across 3366 samples within 1 assay
## Active assay: Spatial (36601 features, 2000 variable features)
##  3 layers present: counts, data, scale.data
##  1 spatial field of view present: slice2
```

#4. Merge the control & AD

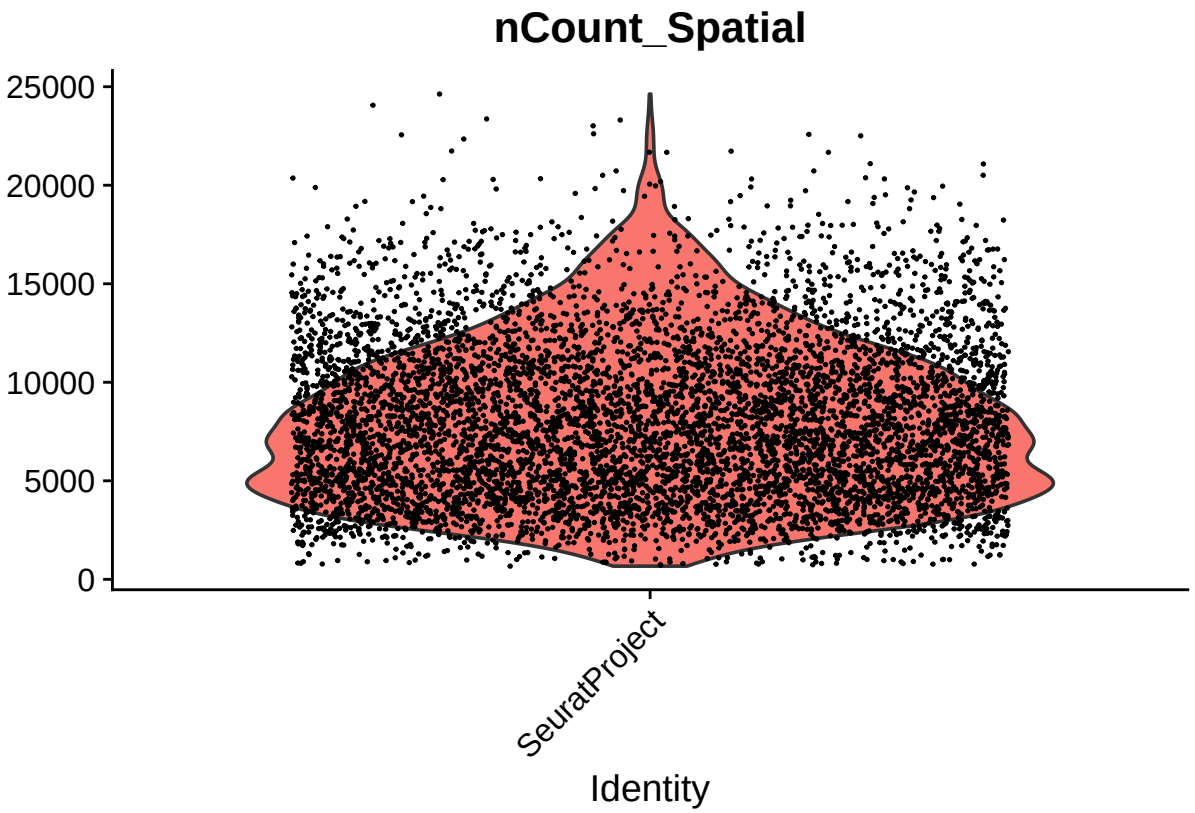
```
#Merging Slices
brain.merge <- merge(control, AD, project = "brain")
```

```
## Warning: Some cell names are duplicated across objects provided. Renaming to
## enforce unique cell names.
```

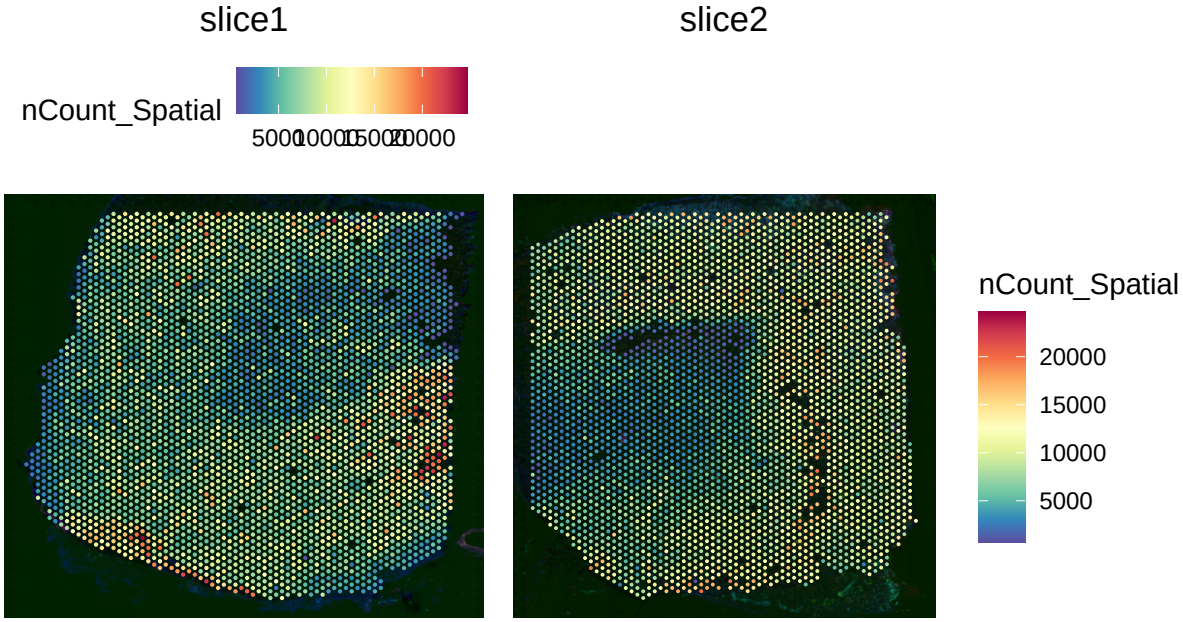
```
#VariableFeatures(brain.merge) <- c(VariableFeatures(control), VariableFeatures(AD))
#Re-Scale Data
all.genes <- rownames(brain.merge)
brain.merge <- ScaleData(brain.merge, features = all.genes)
```

```
## Centering and scaling data matrix
```

```r
#Visualize Results
plot1 <- VlnPlot(brain.merge, features = "nCount_Spatial", pt.size = 0.1) + NoLegend()
plot2 <- SpatialFeaturePlot(brain.merge, features = "nCount_Spatial") + theme(legend.position = "right")
plot1
```

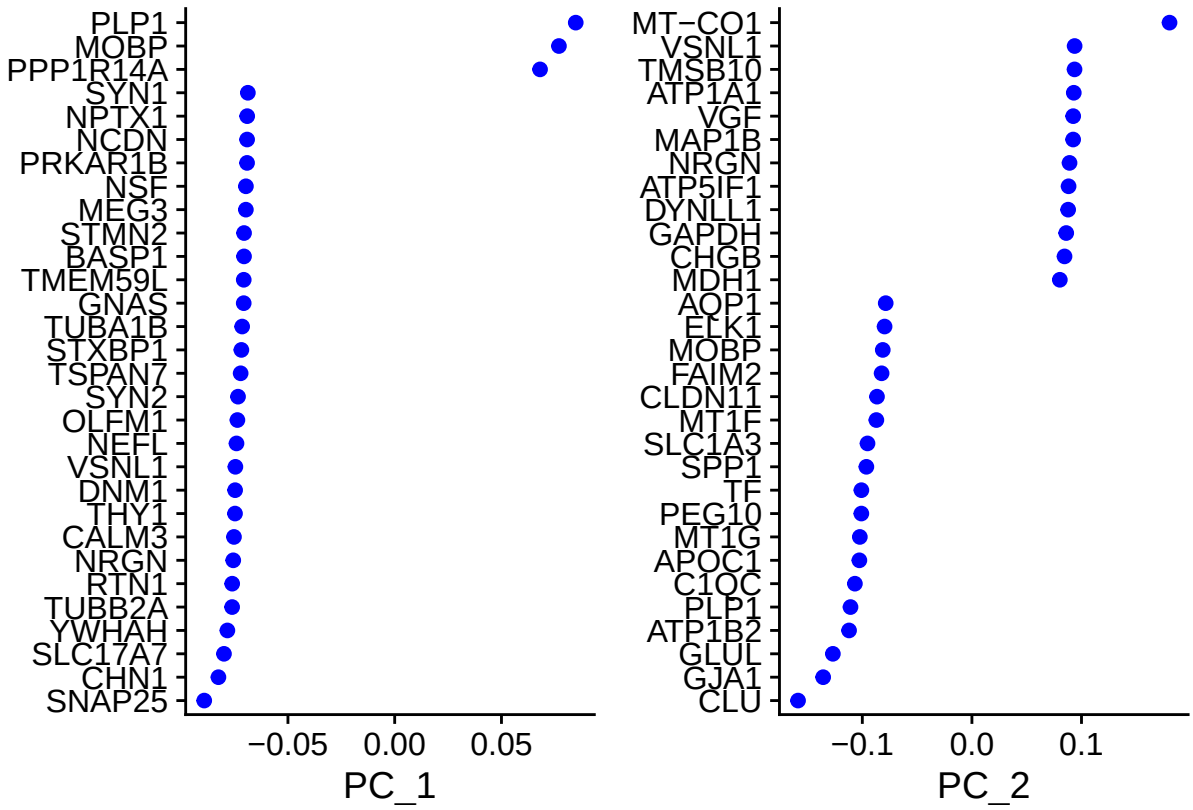



```r
plot2
```

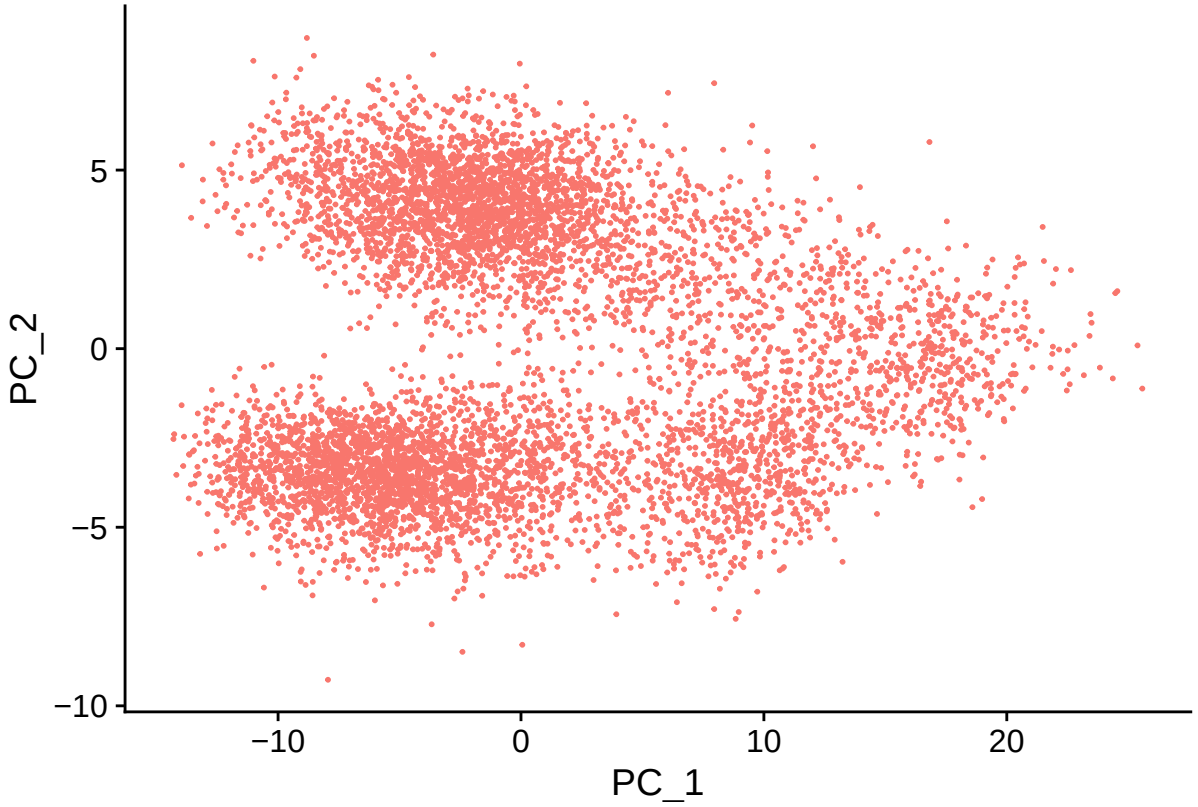


#5 # Dimensionality reduction, clustering, and visualization

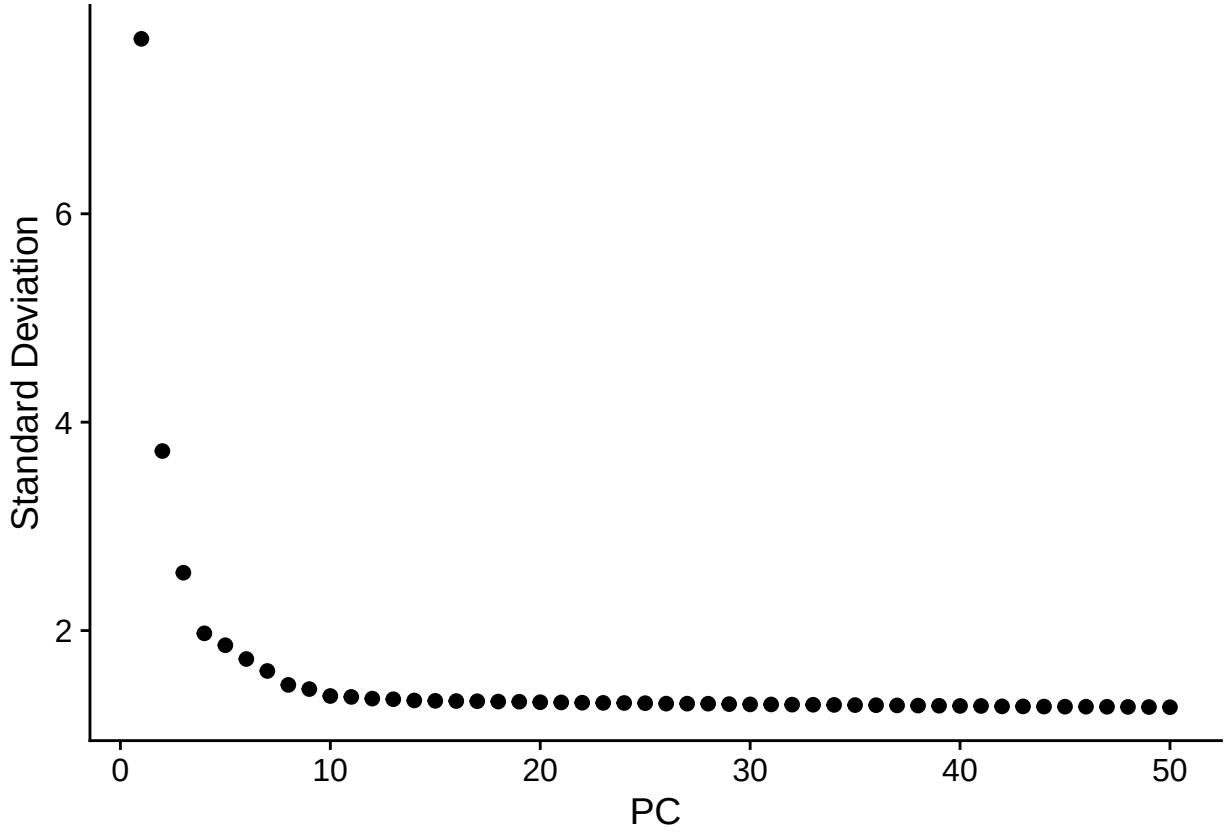
```
#PCA
brain.merge <- RunPCA(brain.merge, assay = "Spatial", verbose = FALSE)
VizDimLoadings(brain.merge, dims = 1:2, reduction = "pca")
```




```
DimPlot(brain.merge, reduction = "pca") + NoLegend()
```




```
ElbowPlot(brain.merge, ndims = 50)
```



```
#Clustering
brain.merge <- FindNeighbors(brain.merge, reduction = "pca", dims = 1:30)
```

```
## Computing nearest neighbor graph
```

```
## Computing SNN
```

```
brain.merge <- FindClusters(brain.merge, resolution = 1, verbose = FALSE)
brain.merge <- RunUMAP(brain.merge, reduction = "pca", dims = 1:30)
```

```
## Warning: The default method for RunUMAP has changed from calling Python UMAP via reticulate to the R
## To use Python UMAP via reticulate, set umap.method to 'umap-learn' and metric to 'correlation'
## This message will be shown once per session
```

```
## 13:10:59 UMAP embedding parameters a = 0.9922 b = 1.112
```

```
## 13:10:59 Read 7023 rows and found 30 numeric columns
```

```
## 13:10:59 Using Annoy for neighbor search, n_neighbors = 30
```

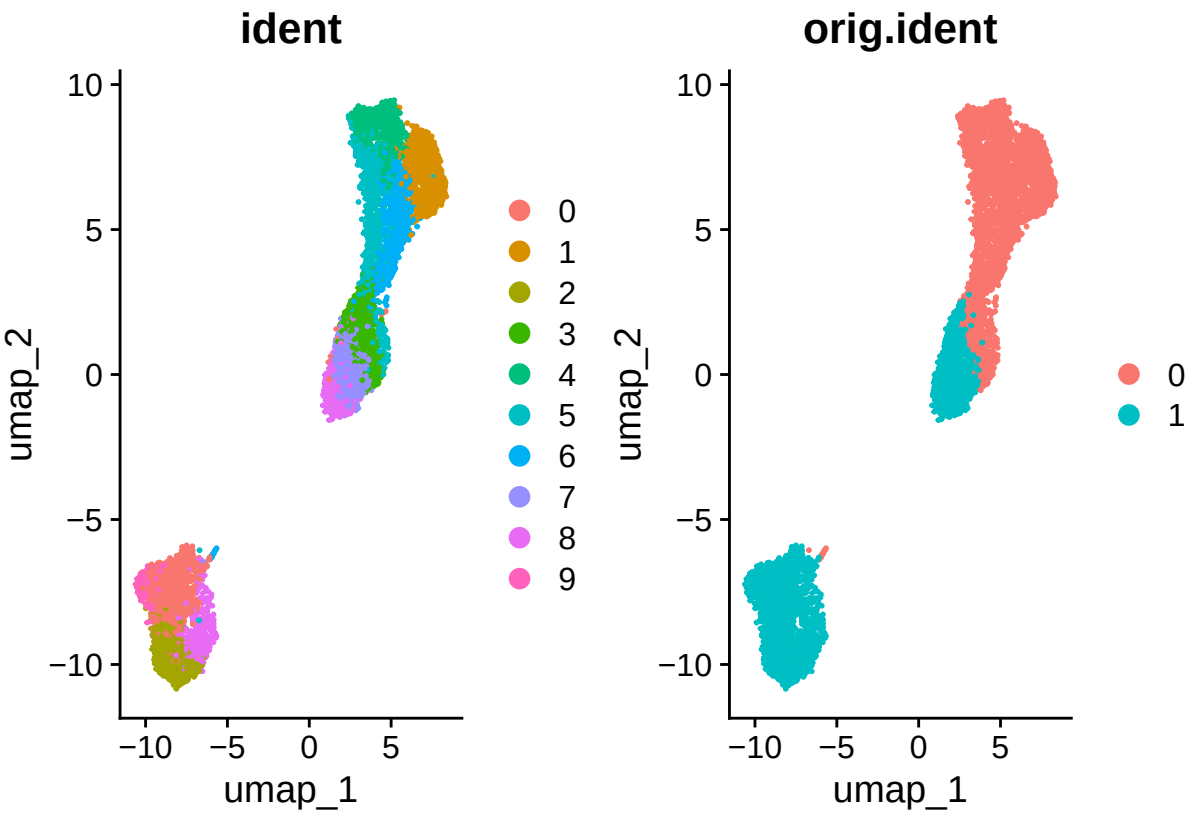
```
## 13:10:59 Building Annoy index with metric = cosine, n_trees = 50
```

```
## 0%   10   20   30   40   50   60   70   80   90   100%
```

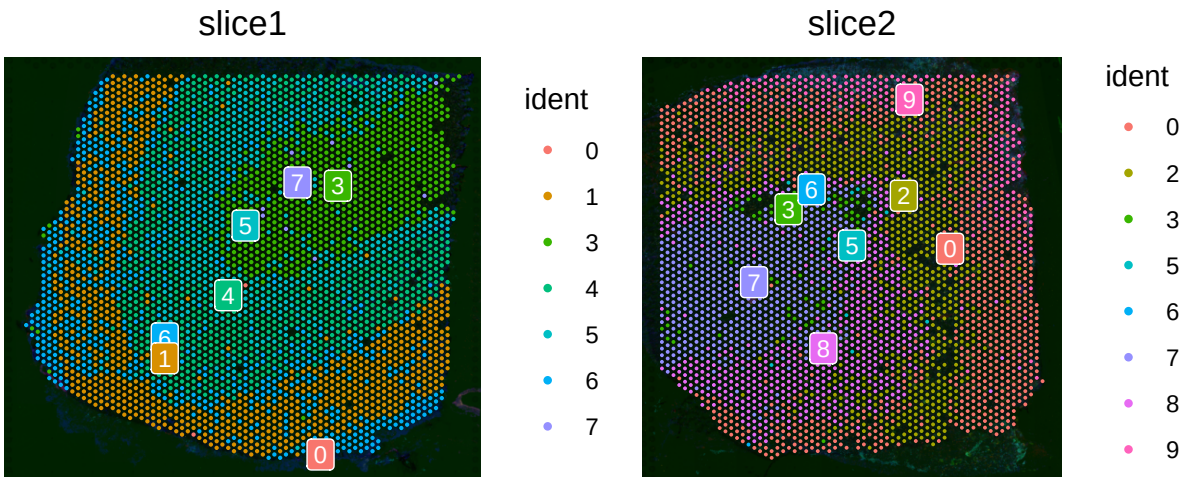
```
## [----|----|----|----|----|----|----|----|----|----|
```

```
## **************************************************|
## 13:11:00 Writing NN index file to temp file /var/folders/rd/nx5v0drj2lq138sbjrd0rb_m0000gn/T//RtmpFBI
## 13:11:00 Searching Annoy index using 1 thread, search_k = 3000
## 13:11:01 Annoy recall = 100%
## 13:11:01 Commencing smooth kNN distance calibration using 1 thread with target n_neighbors = 30
## 13:11:01 Initializing from normalized Laplacian + noise (using RSpectra)
## 13:11:01 Commencing optimization for 500 epochs, with 339508 positive edges
## 13:11:01 Using rng type: pcg
## 13:11:07 Optimization finished
```

```
DimPlot(brain.merge, reduction = "umap", group.by = c("ident", "orig.ident"))
```




```
SpatialDimPlot(brain.merge, label = TRUE, label.size = 3)
```



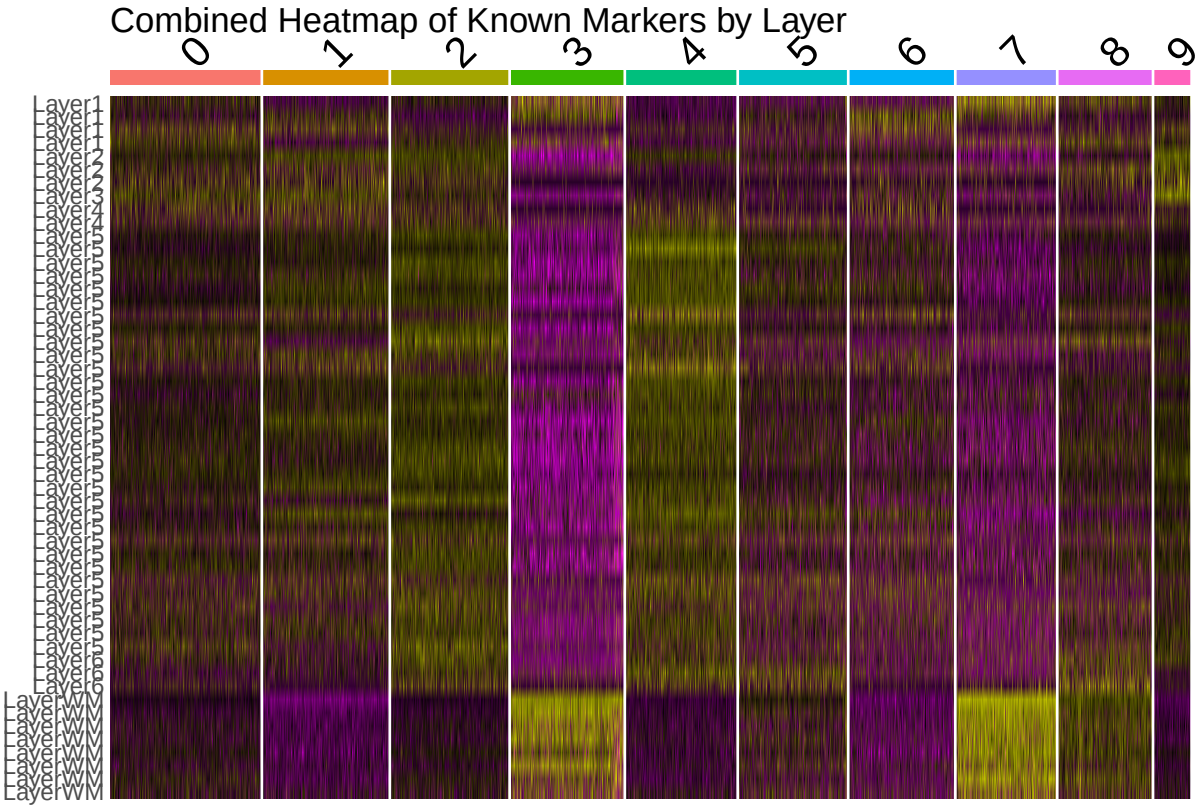
#6 Identifying Cortical Layers by Heatmpap of canonical markers

```
# Define the list of known markers organized by layer
known_markers_list <- list(
    Layer1 = c("SPARC", "MALAT1", "CXCL14", "ADIRF"),
    Layer2 = c("ENC1", "HPCAL1", "CALB2"),
    Layer3 = c("HOPX"),
    Layer4 = c("RORB", "NUAK1"),
    Layer5 = c("TUBA1B", "TMSB10", "SYT1", "STMN2", "MAP1B", "SNAP25", "PCP4", "UCHL1", "CLSTN2", "SNCG
    Layer6 = c("MAP2K1", "DIRAS2", "KRT17"),
    LayerWM = c("MBP", "MOBP", "CLDND1", "BCAS1", "MTURN", "PAQR6", "HIPK2", "DYNC1LI2")
)
```

```r
# Combine all markers into a single vector and create a corresponding layer vector
combined_markers <- unlist(known_markers_list)
layer_labels <- rep(names(known_markers_list), times = sapply(known_markers_list, length))
# Ensure combined markers exist in the Seurat object
valid_markers <- combined_markers[combined_markers %in% rownames(brain.merge)]
valid_layer_labels <- layer_labels[combined_markers %in% rownames(brain.merge)]
# Check if we have valid markers before plotting
if (length(valid_markers) > 0) {
    # Create a named vector for y-axis annotations
    names(valid_layer_labels) <- valid_markers

    # Generate the heatmap
    heatmap_plot <- DoHeatmap(brain.merge, features = valid_markers) +
        NoLegend() +
        scale_y_discrete(labels = valid_layer_labels) +  # Update y-axis labels
        ggtitle("Combined Heatmap of Known Markers by Layer")  # Title for the heatmap

    # Display the heatmap
    print(heatmap_plot)
} else {
    cat("No valid markers found for heatmap generation.\n")
}
```



#7. Subset the Data by Region

```r
#Subset out anatomical regions
## Subset to clusters of interest
WM <- subset(brain.merge, idents = c(3, 7))
```

```
## Warning: Not validating Centroids objects
## Not validating Centroids objects
```

```
## Warning: Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
```

```
## Warning: Not validating Seurat objects
```

```
## Warning: Not validating Centroids objects
## Not validating Centroids objects
```

```
## Warning: Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
## Not validating FOV objects
```

```
## Warning: Not validating Seurat objects
```

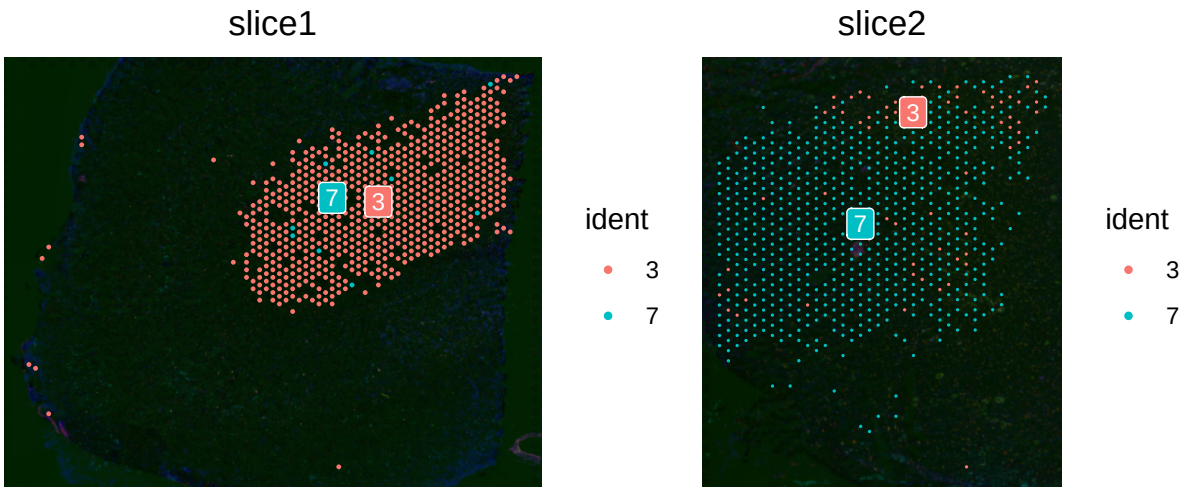
```
# Extract and attach spatial coordinates from image 1
centroids <- WM[["slice1"]]@boundaries$centroids
coords <- setNames(as.data.frame(centroids@coords), c("x", "y"))
rownames(coords) <- centroids@cells
WM$x <- coords[colnames(WM), "x"]
WM$y <- coords[colnames(WM), "y"]

# After subsetting, we renormalize WM
WN <- NormalizeData(WM, normalization.method = "LogNormalize", scale.factor = 10000)
```

```
## Normalizing layer: counts.1
```

```
## Normalizing layer: counts.2
```

```
WM <- RunPCA(WM, verbose = FALSE)
SpatialDimPlot(WM, label = TRUE, label.size = 3)
```



#8 WM DEGs

```r
# Join Layers
WM <- JoinLayers(WM)

# Set the identity classes based on the 'orig.ident' column
WM <- SetIdent(WM, value = WM$orig.ident)


# Compare cells with orig.ident values of 0 and 1
WM.AD.markers <- FindMarkers(WM, ident.1 = 0, ident.2 = 1)
```

```
## For a (much!) faster implementation of the Wilcoxon Rank Sum Test,
## (default method for FindMarkers) please install the presto package
## --------------------------------------------
## install.packages('devtools')
## devtools::install_github('immunogenomics/presto')
## --------------------------------------------
## After installation of presto, Seurat will automatically use the more
## efficient implementation (no further action necessary).
## This message will be shown once per session
```

```r
# View the results
head(WM.AD.markers)
```

```
##                  p_val avg_log2FC pct.1 pct.2     p_val_adj
## MTRNR2L1 8.821611e-251 -6.6071712 0.120 0.999 3.228798e-246
## MTRNR2L8 3.526000e-196  1.9377902 0.991 0.955 1.290551e-191
## RPS26    1.161904e-182  2.7236926 0.959 0.460 4.252685e-178
## MT-CO2   3.698496e-136  0.5711470 1.000 1.000 1.353686e-131
## S100B    8.613357e-131  1.4885077 0.983 0.876 3.152575e-126
## RPL41    1.171041e-130  0.8657436 0.997 0.982 4.286126e-126
```