Session 10 - Assignment 2

Please Write Your Name

2023-03-15

# Assignment 2

You have just joined a lab that is trying to reproduce the single cell RNA seq results published by Nowakowski et al in 2017 studying the fetal brain [ref](https://www.science.org/doi/10.1126/science.aap8809). You were given the R markdown code to continue the work from a previous post-doc and discover that only some of the steps work. Everything else either doesn’t make sense, or doesn’t run. Follow the tasks below, and refer to the materials from Session 9, or the original [Seurat guided clustering tutorial](https://satijalab.org/seurat/articles/pbmc3k_tutorial.html) to correct and clean this R markdown file.

Create groups with 3-5 people and complete the tasks described below.

Turning in the assignment: Please knit the Rmd document into a word document and upload the file into the assignments page in bruin learn. **DO NOT FORGET** to add your name next to author at the top of the document. **Be sure that the code you write is visible** in the knitted document. If the document does not knit and you are unable to identify why, please submit the document as an .Rmd file.

## Task 1: Load Libraries

Post-doc notes: So I’m trying to load all the packages for this analysis and I can’t seem to get the Seurat package to work. None of the commands are working and for whatever reason half my packages aren’t loading. Correct the code in the chunk below, and include notes for each of the packages loaded.

rm(list=ls())  
knitr::opts\_chunk$set(echo = TRUE)  
library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.4.1 ✔ purrr 1.0.1  
## ✔ tibble 3.2.0 ✔ dplyr 1.1.0  
## ✔ tidyr 1.3.0 ✔ stringr 1.5.0  
## ✔ readr 2.1.3 ✔ forcats 0.5.2  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(ggplot2)  
library(Seurat)

## Attaching SeuratObject

library(data.table)

##   
## Attaching package: 'data.table'  
##   
## The following objects are masked from 'package:dplyr':  
##   
## between, first, last  
##   
## The following object is masked from 'package:purrr':  
##   
## transpose

library(patchwork)

**Describe what you had to correct in order to get the setup to work:**

* ggplot is missing a 2 in the library name
* Seurat is not loaded
* there is a typo in patchwork

## Task 2: Import the data (1 point)

Post-doc notes: Every time I run this chunk it seems like all the data loads for a second and then just disappears. Why the heck does this keep happening?

mat <- fread("Datasets/NowakSubsampled.tsv")  
meta <- read.table("meta.tsv", header=T, sep="\t", as.is=T, row.names=1)  
genes = mat[,1][[1]]  
genes = gsub(".+[|]", "", genes)  
mat = data.frame(mat[,-1], row.names=genes)  
Nowak <- CreateSeuratObject(counts = mat, project = "brain", meta.data=meta)

## Warning in CreateSeuratObject.default(counts = mat, project = "brain",  
## meta.data = meta): Some cells in meta.data not present in provided counts  
## matrix

## Warning: Feature names cannot have underscores ('\_'), replacing with dashes  
## ('-')

**Describe what you had to correct in order to get the data import to work:**

* rm(list=ls()) shouldn’t be added after you load in your data set. Removing that line will get it to work

## Task 3: Quality Control Part 1 (1 point)

Post-doc notes: I’m pretty sure at some point I was able to get this to work. I’m getting some weird “unexpected assignment” error in the first step, and some “no slot” error in the second step.

# Calculate the percent of Mitochondrial features.  
Nowak[["percent.mt"]] <- PercentageFeatureSet(Nowak, pattern = "^MT-")  
  
# Show QC metrics for the first 5 cells  
head(Nowak@meta.data, 5)

## orig.ident nCount\_RNA nFeature\_RNA WGCNAcluster Name Age\_in\_Weeks  
## O2.F6 brain 976766.4 2586 nEN-early2 Sample3 18.5  
## S91.F6 brain 973139.6 2138 IN-STR Sample33 16.0  
## S94.A5 brain 984903.6 3194 EN-V1-2 Sample37 13.0  
## O11.H4 brain 982552.7 1606 nIN5 Sample6 16.0  
## S140.G12 brain 970162.6 2653 nEN-late Sample42 20.0  
## RegionName Laminae Area percent.mt  
## O2.F6 Cortex VZ V1 0  
## S91.F6 Cortex 0 0 0  
## S94.A5 Cortex All V1 0  
## O11.H4 GE 0 MGE 0  
## S140.G12 Cortex GZ 0 0

**Describe what you had to correct in order to get the data filtering to work:**

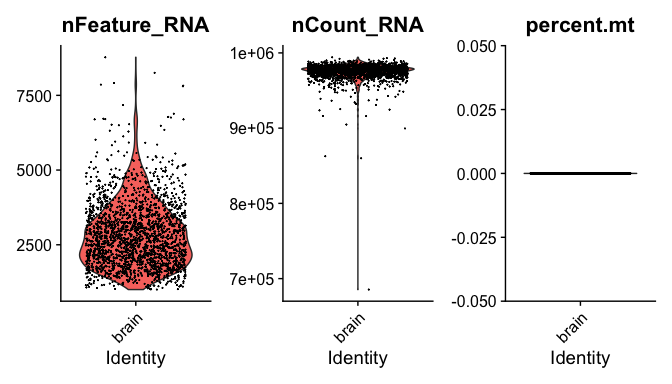
* A bracket is missing from the first step
* there is an extra t in the meta.data line

## Task 4: Quality Control Part 2 (1 point)

Post-doc notes: I was getting the violin plot command to work, but for some reason the number of Features and Counts in the data set are not showing up. I know the step was working at some point because someone told me that there were no mitochondrial genes showing up in the data set, and that’s the only plot I can see.

# Visualize QC metrics as a violin plot  
VlnPlot(Nowak, features = c("nFeature\_RNA", "nCount\_RNA", "percent.mt"), ncol = 3)

## Warning in SingleExIPlot(type = type, data = data[, x, drop = FALSE], idents =  
## idents, : All cells have the same value of percent.mt.



**Correct the errors in the commands, and describe the distribution of the number of features and counts in the data set:**

* the parameters Features\_RNA, and Count\_RNA are missing “n” in the character string

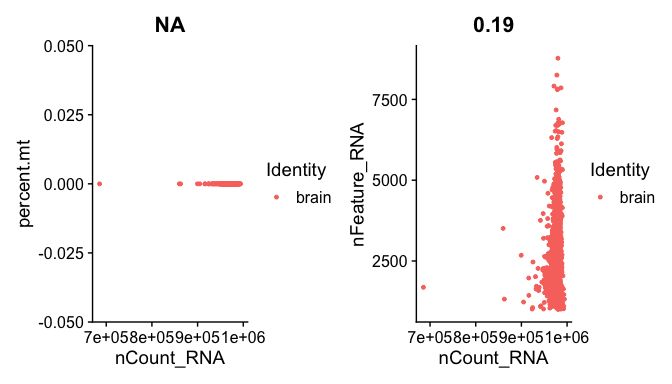
## Task 5: Quality Control Part 3 (1 point)

Post-doc notes: Okay as soon as I thought things were working for me everything is a mess again. I was able to get the scatter plots to work, but now when I filter to create my filtered object NowakFilter. this thing tells me I have no cells. What is happening?

# FeatureScatter is typically used to visualize feature-feature relationships, but can be used  
# for anything calculated by the object, i.e. columns in object metadata, PC scores etc.  
mitoscatter <- FeatureScatter(Nowak, feature1 = "nCount\_RNA", feature2 = "percent.mt")

## Warning in cor(x = data[, 1], y = data[, 2]): the standard deviation is zero

featscatter <- FeatureScatter(Nowak, feature1 = "nCount\_RNA", feature2 = "nFeature\_RNA")  
mitoscatter + featscatter



# using subset, you complete the filtering step  
NowakFilter <- subset(Nowak, subset = nFeature\_RNA > 200 & nFeature\_RNA < 2500 & percent.mt < 5)

**Describe what you had to correct in the subset command, and the rationale for subsetting in that way.**

* the subset step is not correct, the number of features should be greater than 200 and less than 2500.
* this filtering is used to remove dead cells, and remove doublets

## Task 6: Normalize Data and Find Variable Features (1 point)

Post-doc notes: I’m just going to run the NormalizeData command with no extra arguments so I don’t mess it up. Looks like it works as expected…

NowakFilter <- NormalizeData(NowakFilter)

Post-doc notes: The thing is… after I find variable features, all the gene names seem to overlap. I’m trying to filter on the top 5 genes, but for some reason the head() command is giving me a ton of extra names.

On top of that, all the gene names are stacked on top of each other in the plot… I can’t read any of this. How is anyone supposed to get this to work?

NowakFilter <- FindVariableFeatures(NowakFilter, selection.method = "vst", nfeatures = 2000)  
  
# Identify the 5 most highly variable genes  
top5 <- head(VariableFeatures(NowakFilter), 5)  
top5

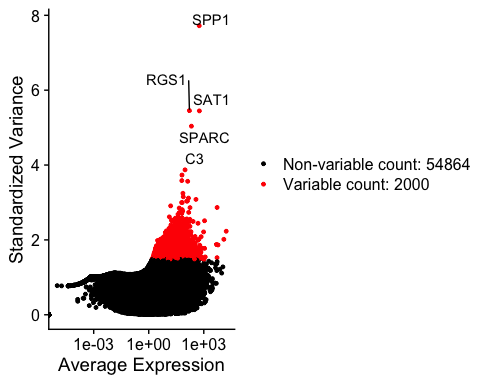
## [1] "SPP1" "RGS1" "SAT1" "SPARC" "C3"

# plot variable features with labels  
plot1 <- VariableFeaturePlot(NowakFilter)  
plot2 <- LabelPoints(plot = plot1, points = top5, repel = T)

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

plot2

## Warning: Transformation introduced infinite values in continuous x-axis



**Correct the issue in the head(), and plotting commands. Describe what you corrected and list the 5 genes that contribute most to variability in the dataset:**

* the head command contains the top 20 genes and not the top 5 genes.
* the repel feature should = T, not F

## Task 7: Scale the data and run PCA (1 point)

Post-doc notes: Okay, keeping it simple worked before, I’ll do the same and hope for the best.

NowakFilter <- ScaleData(NowakFilter)

## Centering and scaling data matrix

Post-doc notes: YES! The scaling seems to work. Now I can run the PCA

NowakFilter <- RunPCA(NowakFilter, features = VariableFeatures(object = NowakFilter))

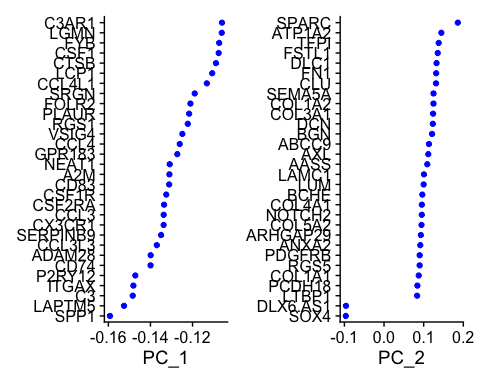
## PC\_ 1   
## Positive: SOX4, MAP1B, DPYSL2, DLX6.AS1, RBFOX2, PTPRD, ERBB4, GAD1, SLAIN1, POU3F2   
## MN1, LINC01102, CCND2, UNC79, ZNF711, SACS, INA, FZD3, ST8SIA5, ST6GAL2   
## SUGT1, GRIA1, UNC5D, ACVR2A, XPR1, CNR1, ENC1, PCDH9, KIFAP3, FGF12   
## Negative: SPP1, LAPTM5, C3, ITGAX, P2RY12, CD74, ADAM28, CCL3L3, SERPINB9, CX3CR1   
## CCL3, CSF2RA, CSF1R, CD83, A2M, NEAT1, GPR183, CCL4, VSIG4, RGS1   
## PLAUR, FOLR2, SRGN, CCL4L1, LCP1, CTSB, CSF1, FYB, LGMN, C3AR1   
## PC\_ 2   
## Positive: SPARC, ATP1A2, TFPI, FSTL1, DLC1, FN1, CLU, SEMA5A, COL1A2, COL3A1   
## DCN, BGN, ABCC9, AXL, AASS, LAMC1, LUM, BCHE, COL4A1, NOTCH2   
## COL5A2, ARHGAP29, ANXA2, PDGFRB, RGS5, COL1A1, PCDH18, LTBP1, HBEGF, EBF1   
## Negative: SOX4, DLX6.AS1, ERBB4, GAD1, MAP1B, MAF, ST8SIA5, SLAIN1, SPP1, LAPTM5   
## C3, CD74, CSF1R, P2RY12, ITGAX, CX3CR1, INA, ADAM28, GRIA1, FOLR2   
## CSF2RA, VSIG4, GPR183, PROX1, CCL3L3, RBFOX2, CCL3, PRKCA, CCL4, FYB   
## PC\_ 3   
## Positive: DCN, COL3A1, BGN, FN1, DLC1, ABCC9, LAMC1, LUM, TFPI, COL4A1   
## SPARC, FBN1, EBF1, CDH6, RBMS3, COL1A1, LRRC32, ERBB4, ARHGAP29, MEF2C   
## MAF, COL1A2, OLFML3, ARHGDIB, PDGFRB, GUCY1B3, UACA, RGS5, RBFOX2, COL5A2   
## Negative: TOP2A, CENPF, MKI67, CENPE, KIF23, KIF15, ASPM, HJURP, CCNB1, SPAG5   
## TTK, HIST1H4C, KPNA2, CDCA8, TROAP, CEP152, DEPDC1B, MELK, KNTC1, NCAPD2   
## CCND2, HELLS, FANCI, DTL, MID1, IL33, TOX3, RTKN2, SKA3, EZH2   
## PC\_ 4   
## Positive: ENC1, PTPRZ1, LINC01102, POU3F2, SLC24A2, KIAA1456, UNC5D, ST8SIA1, PTPRD, ZFHX4   
## LIMCH1, CNR1, ITGB8, NTRK3, FAM126A, KCNN3, PDE1C, THBS1, SETD7, BHLHE22   
## SEMA4D, EPHB1, PRSS12, SEMA3A, SAT1, CCND2, GRIA3, HS6ST3, DAB1, GAREM   
## Negative: DLX6.AS1, ERBB4, GAD1, MAF, ST8SIA5, EFNA5, SLAIN1, SACS, ZSWIM5, TOX3   
## TOP2A, ASPM, SOX4, COL1A2, EGR1, BMP3, STXBP5L, FOS, GRIA4, MKI67   
## RBMS3, PRKCA, CENPE, CENPF, PROX1, HJURP, DCN, INA, FBN1, SLC4A4   
## PC\_ 5   
## Positive: ENC1, POU3F2, UNC5D, SOX4, THBS1, COL1A2, COL3A1, SLC24A2, DCN, BGN   
## ABCC9, DLC1, FN1, ST8SIA1, COL4A1, SETD7, LUM, CNR1, FAM126A, CDH6   
## RGS5, COL5A2, KIAA1456, FHDC1, OLFML3, BHLHE22, RBMX, LAMC1, HUNK, CCND2   
## Negative: PDGFRA, PCDH15, LUZP2, CNTN1, UGT8, APOD, CSGALNACT1, NTNG1, TRIL, LRRC4C   
## PMP2, SLC35F1, PRCP, PCDH17, SULF2, TNC, MMP16, ETV5, SEMA5A, AC144652.1   
## PLEKHH2, AC012309.5, FAM84A, CSMD3, NTRK3, PTPRT, FAM107A, P2RX7, FTSJ2, AQP4

Post-doc notes: Great. I was able to create the plots for the next steps, but I don’t remember how to interpret these…

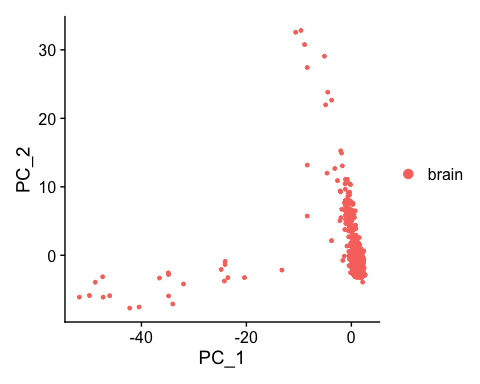
# Examine and visualize PCA results a few different ways  
print(NowakFilter[["pca"]], dims = 1:5, nfeatures = 5)

## PC\_ 1   
## Positive: SOX4, MAP1B, DPYSL2, DLX6.AS1, RBFOX2   
## Negative: SPP1, LAPTM5, C3, ITGAX, P2RY12   
## PC\_ 2   
## Positive: SPARC, ATP1A2, TFPI, FSTL1, DLC1   
## Negative: SOX4, DLX6.AS1, ERBB4, GAD1, MAP1B   
## PC\_ 3   
## Positive: DCN, COL3A1, BGN, FN1, DLC1   
## Negative: TOP2A, CENPF, MKI67, CENPE, KIF23   
## PC\_ 4   
## Positive: ENC1, PTPRZ1, LINC01102, POU3F2, SLC24A2   
## Negative: DLX6.AS1, ERBB4, GAD1, MAF, ST8SIA5   
## PC\_ 5   
## Positive: ENC1, POU3F2, UNC5D, SOX4, THBS1   
## Negative: PDGFRA, PCDH15, LUZP2, CNTN1, UGT8

VizDimLoadings(NowakFilter, dims = 1:2, reduction = "pca")



DimPlot(NowakFilter, reduction = "pca")



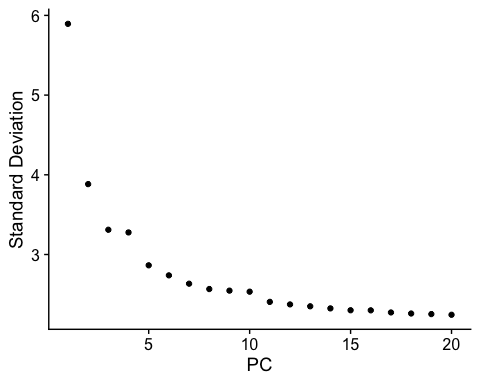
**Describe how you would interpret the plots created by VizDimLoadings() and DimPlot():**

* The VizDimLoadings plot shows the amount each gene contributs to its respective principal component
* The DimPlot shows the top two PCs similar to a standard identity plot

## Task 8: Interpreting the Elbow Plot

Post-doc notes: I have the jackstraw plots and dimension heatmaps saved somewhere else, so I don’t need to run them again. I remember the ElbowPlot is supposed to help me figure out something I can use for clustering, but I cant remember what…

ElbowPlot(NowakFilter)



**Describe what the elbow plot is showing, and how you can use it to decide how to run the clustering step:**

* The elbow plot is similar to a scree plot. Shows how much each PC contributes to the overall variability in the dataset.
* You can use this plot to decide how many principal components you need to include in the clustering step

## Task 9: Create a UMAP diagram with correct number of clusters (1 point)

Post-doc Notes: Someone told me that there were supposed to be 7 clusters in this subset of the data set, but for some reason my plot has more clusters and none of them make sense. I can’t figure out what the issue is.

NowakFilter <- FindNeighbors(NowakFilter, dims = 1:10)

## Computing nearest neighbor graph

## Computing SNN

NowakFilter <- FindClusters(NowakFilter, resolution = 0.5)

## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck  
##   
## Number of nodes: 874  
## Number of edges: 32554  
##   
## Running Louvain algorithm...  
## Maximum modularity in 10 random starts: 0.8046  
## Number of communities: 7  
## Elapsed time: 0 seconds

# If you haven't installed UMAP, you can do so via reticulate::py\_install(packages = 'umap-learn')  
# I'm pretty sure this step is working  
NowakFilter <- RunUMAP(NowakFilter, dims = 1:10)

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

## 14:37:24 UMAP embedding parameters a = 0.9922 b = 1.112

## 14:37:24 Read 874 rows and found 10 numeric columns

## 14:37:24 Using Annoy for neighbor search, n\_neighbors = 30

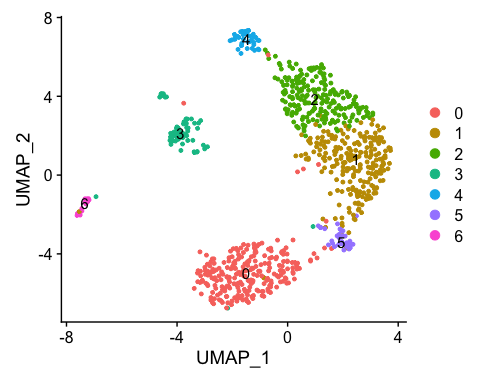
## 14:37:24 Building Annoy index with metric = cosine, n\_trees = 50

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

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

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*|  
## 14:37:24 Writing NN index file to temp file /var/folders/0t/rz851wzj23z4chccf3y4q48m3b3x0m/T//RtmpLJzb9E/file1859d3936f035  
## 14:37:24 Searching Annoy index using 1 thread, search\_k = 3000  
## 14:37:24 Annoy recall = 100%  
## 14:37:24 Commencing smooth kNN distance calibration using 1 thread with target n\_neighbors = 30  
## 14:37:25 Initializing from normalized Laplacian + noise (using irlba)  
## 14:37:25 Commencing optimization for 500 epochs, with 34800 positive edges  
## 14:37:26 Optimization finished

# I'm pretty sure DimPlot is what I use to view the UMAP  
DimPlot(NowakFilter, reduction = "umap", label = T)



**Describe the problem in the clustering steps and create a umap with clusters that group together.**

* The FindNeighbors step uses only one dimension to find the neighbors. You have to use all the dimensions you identify in the ElbowPlot in order to properly cluster

## Task 10: Create a t-SNE plot (1 point)

Post-doc notes: I can’t find the code to create the tSNE plot in the Seurat object. We need the t-SNE for our own analysis.

**Search the web for the Seurat command used to create a t-SNE plot. Write the code in the chunk below:**

# run the t-SNE command using the first 10 dimensions  
NowakFilter <- RunTSNE(NowakFilter, dims = 1:10)  
  
# I'm pretty sure I'm supposed to use DimPlot() to view the tsne plot  
DimPlot(NowakFilter, reduction = "tsne")

