Assingment3_Gabriel

Install any necessary packages:

```
#install.packages("matrixStats")
#install.packages("mclust")
```

Exercise 1

Load the processed data into R:

```
PATH_TO_FILE = "gene.tsv"
df = read.delim(PATH_TO_FILE, sep= "\t", header = TRUE)
```

Exercise 2

Subset data to the 5000 most variable genes

```
library(matrixStats)
gene_var <- rowVars(as.matrix(df[,-1]))
df_top5000 <- df[order(gene_var, decreasing = TRUE)[1:5000], ]
```

I Will be using *Gaussian mixture models* as my clustering algorithm

```
library(mclust)
```

```
## Package 'mclust' version 6.1.1
## Type 'citation("mclust")' for citing this R package in publications.
```

```
# we need rows as samples to apply GMM, so we will transpose
df_top5000_t = t(df_top5000[,-1])

# Run GMM
gmm_fit = Mclust(df_top5000_t)

# Check model summary
summary(gmm_fit)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 9 components:
##
##  log-likelihood   n    df      BIC      ICL
##        -2882538 307 50016 -6051510 -6051510
##
## Clustering table:
##  1  2  3  4  5  6  7  8  9
## 68 67 10 75 57  6  6 12  6
```

As we can see in the output above, the chosen number of clusters is 9. We can alter this by changing the number of clusters the function tries, which I will do below:

```
# 1:5
gmm_alt_fit1 = Mclust(df_top5000_t, G = 1:5)
summary(gmm_alt_fit1)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 5 components:
##
##  log-likelihood   n    df      BIC      ICL
##        -2989571 307 30008 -6150993 -6150993
##
## Clustering table:
##  1  2  3  4  5
## 73 72 75 63 24
```

```
# 9:15
gmm_alt_fit2 = Mclust(df_top5000_t, G = 9:15)
summary(gmm_alt_fit2)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 13 components:
##
##  log-likelihood   n    df      BIC      ICL
##        -2798885 307 70024 -5998787 -5998787
##
## Clustering table:
##  1  2  3  4  5  6  7  8  9 10 11 12 13
## 68 67 10 48 27 17  6  6 17 12 18  6  5
```

From the output above, we see that if we experiment only with 1 to 5 clusters, 5 is the number chosen by the function. However, if we experiment with 9 to 15, with 9 being the original number of clusters chosen, we get 13 as being the best fit. This leads me to think that 13 is a better number of clusters than 9, as it subdivides the data further with and maximizes BIC.

Now we are asked to rerun the clustering method using 10, 100, 1000, and 10000 genes. Let's start with 10:

```
# run gmm on top 10
gmm_fit_10 = Mclust(df_top5000_t[,1:10], G = 1:15)
# Check model summary
summary(gmm_fit_10)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEV (ellipsoidal, equal shape) model with 9 components:
##
##  log-likelihood   n  df       BIC       ICL
##       -4502.476 307 521 -11988.64 -11988.97
##
## Clustering table:
##  1  2  3  4  5  6  7  8  9
## 22 92 21 10 19 41 15 64 23
```

Now 100:

```
# run gmm on top 100
gmm_fit_100 = Mclust(df_top5000_t[,1:100], G = 1:15)
# Check model summary
summary(gmm_fit_100)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EEE (ellipsoidal, equal volume, shape and orientation) model with 15
## components:
##
##  log-likelihood   n   df       BIC       ICL
##        -57803.6 307 6564 -153198.2 -153198.2
##
## Clustering table:
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
## 22  4 36 15  5  5 32 21  5 19 21 30  5 76 11
```

Now 1000:

```
# run gmm on top 1000
gmm_fit_1000 = Mclust(df_top5000_t[,1:1000], G = 1:15)
# Check model summary
summary(gmm_fit_1000)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 15 components:
##
##  log-likelihood   n    df      BIC      ICL
##       -683115.4 307 16028 -1458021 -1458021
##
## Clustering table:
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
## 22  9 51 10 53 19 34 22 26  6  6 25 12  6  6
```

Now 10000:

```
# get top 10000
df_top10000 <- df[order(gene_var, decreasing = TRUE)[1:10000], ]
df_top10000_t = t(df_top10000[,-1])

# run gmm on top 10000
gmm_fit_10000 = Mclust(df_top10000_t, G = 1:15)
# Check model summary
summary(gmm_fit_10000)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust VEI (diagonal, equal shape) model with 13 components:
##
##  log-likelihood   n     df      BIC      ICL
##        -4516720 307 140024 -9835335 -9835335
##
## Clustering table:
##  1  2  3  4  5  6  7  8  9 10 11 12 13
## 68 67 10 75  6  6  6 23 11 18  6  6  5
```

We are interested in understanding how the number of genes affects clustering. To determine this, we will perform a chi-squared test on each pair of clustering results (10, 100, 1000, 5000, and 10000 genes):

```
# create a list containing all clustering results
clust_10 = gmm_fit_10$classification
clust_100 = gmm_fit_100$classification
clust_1000 = gmm_fit_1000$classification
clust_5000 = gmm_alt_fit2$classification
clust_10000 = gmm_fit_10000$classification

# compare 10 vs 100
chisq.test(clust_10, clust_100)
```

```
## Warning in chisq.test(clust_10, clust_100): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_10 and clust_100
## X-squared = 1728.7, df = 112, p-value < 2.2e-16
```

```
# compare 10 vs 1000
chisq.test(clust_10, clust_1000)
```

```
## Warning in chisq.test(clust_10, clust_1000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_10 and clust_1000
## X-squared = 1479, df = 112, p-value < 2.2e-16
```

```
# compare 10 vs 5000
chisq.test(clust_10, clust_5000)
```

```
## Warning in chisq.test(clust_10, clust_5000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_10 and clust_5000
## X-squared = 987.06, df = 96, p-value < 2.2e-16
```

```
# compare 10 vs 10000
chisq.test(clust_10, clust_10000)
```

```
## Warning in chisq.test(clust_10, clust_10000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_10 and clust_10000
## X-squared = 921.15, df = 96, p-value < 2.2e-16
```

```
# compare 100 vs 1000
chisq.test(clust_100, clust_1000)
```

```
## Warning in chisq.test(clust_100, clust_1000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_100 and clust_1000
## X-squared = 2420, df = 196, p-value < 2.2e-16
```

```
# compare 100 vs 5000
chisq.test(clust_100, clust_5000)
```

```
## Warning in chisq.test(clust_100, clust_5000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_100 and clust_5000
## X-squared = 1582.4, df = 168, p-value < 2.2e-16
```

```
# compare 100 vs 10000
chisq.test(clust_100, clust_10000)
```

```
## Warning in chisq.test(clust_100, clust_10000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_100 and clust_10000
## X-squared = 1491.6, df = 168, p-value < 2.2e-16
```

```
# compare 1000 vs 5000
chisq.test(clust_1000, clust_5000)
```

```
## Warning in chisq.test(clust_1000, clust_5000): Chi-squared approximation may be
## incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_1000 and clust_5000
## X-squared = 2649.8, df = 168, p-value < 2.2e-16
```

```
# compare 1000 vs 10000
chisq.test(clust_1000, clust_10000)
```

```
## Warning in chisq.test(clust_1000, clust_10000): Chi-squared approximation may
## be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_1000 and clust_10000
## X-squared = 2459.9, df = 168, p-value < 2.2e-16
```

```
# compare 5000 vs 10000
chisq.test(clust_5000, clust_10000)
```

```
## Warning in chisq.test(clust_5000, clust_10000): Chi-squared approximation may
## be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  clust_5000 and clust_10000
## X-squared = 2868.9, df = 144, p-value < 2.2e-16
```