Clustering

K-means

We'll use the tissues gene expression file to look at clustering. There are expressions for 100 genes in 1816 samples. The samples come from 5 different tissues.

```
library(rafalib)
e <- read.delim("data/expr.txt", row.names=1)
tab <- read.delim("data/class_labels.txt", row.names=1)
head(tab) # rows represent samples
```

```
##                          tissue
## GTEX-111CU-1826-SM-5GZYN      3
## GTEX-111FC-0226-SM-5N9B8      3
## GTEX-111VG-2326-SM-5N9BK      3
## GTEX-111YS-2426-SM-5GZZQ      3
## GTEX-1122O-2026-SM-5NQ91      3
## GTEX-1128S-2126-SM-5H12U      3
```

```
head(e)[,1:2] # rows represent genes
```

```
##                    GTEX.111CU.1826.SM.5GZYN GTEX.111FC.0226.SM.5N9B8
## ENSG00000104879.4                -0.4651989               -0.2794045
## ENSG00000143632.10               -0.4107903               -0.1409032
## ENSG00000244734.2                -1.0309454               -0.7426210
## ENSG00000188536.8                -0.9517054               -0.8217659
## ENSG00000206172.4                -0.9177072               -0.8854275
## ENSG00000111245.10               -0.6066508                0.0150647
```

We train the `kmeans` model in the **`stats`** package on the first 5 samples for total cluster number = 5, for a maximum of 10 iterations.

```
set.seed(1)
km <- kmeans(t(e), centers=t(e)[1:5,], iter.max=10)
cbind(cluster=c(1,2,3,4,5), samples=km$size)
```

```
##      cluster samples
## [1,]       1     320
## [2,]       2     430
## [3,]       3     393
## [4,]       4     323
## [5,]       5     350
```

```
table(tissue=tissue, cluster=km$cluster)
```

```
##       cluster
## tissue   1   2   3   4   5
##      0   0 430   0   0   0
##      1 320   0   0   0   0
##      2   0   0   0 323   0
##      3   0   0   0   0 350
##      4   0   0 393   0   0
```

Bayesian Information Criterion

We'll now optimize the number of clusters (k) using Bayesian Information Criterion,

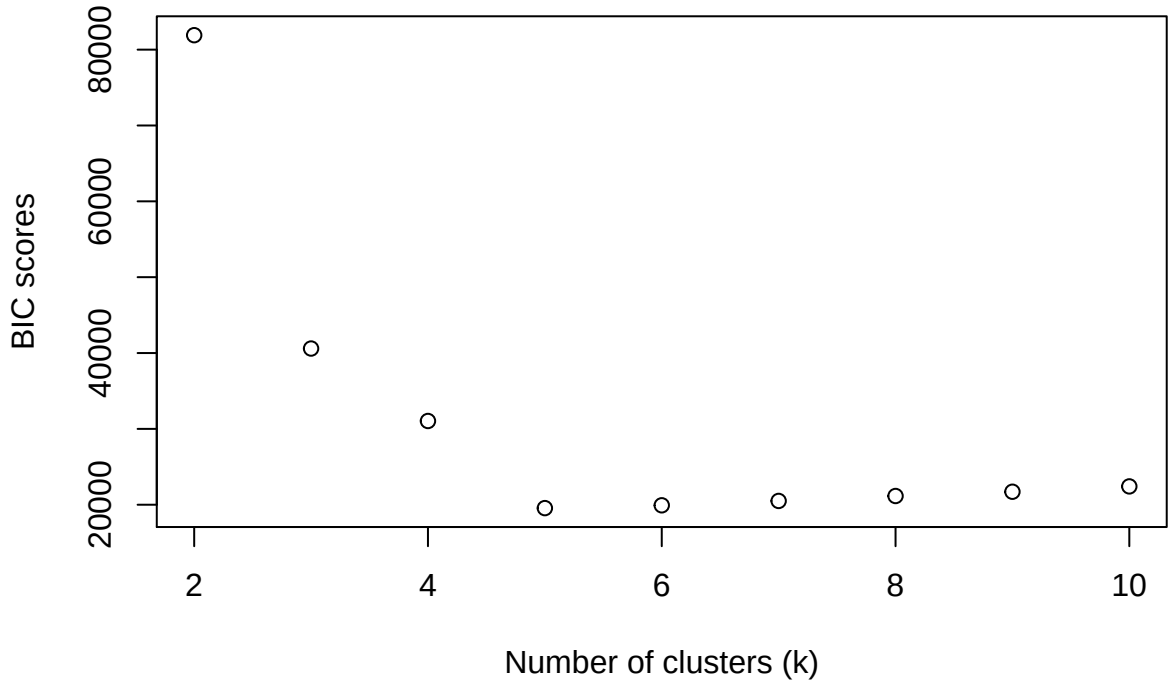
$$BIC = -2 \log \hat{L} + m \log n$$

```
bic <- function(fit) {
  k <- nrow(fit$centers) # number of clusters
  m <- ncol(fit$centers) * k # number of free parameters
  n <- length(fit$cluster) # number of samples
  D <- fit$tot.withinss
  return (D + m*log(n))
}
```

In a k-means model, the number of free parameters m is just the number of clusters k times d-dimesional points. In our case, $d = 100$ (the number of genes), and $k = 5$ (by choice), so $\boldsymbol{m = 500}$.

Now, let us compute the BIC scores for a k-means model trained using $k = 2$ to 10. In each case, the initial centers are the first k points.

```
bic.values <- c()
for (k in 2:10) {
  bic.values[k-1] <- bic(kmeans(t(e), centers=t(e)[1:k,], iter.max=10))
}
par(mfrow = c(1, 1))
plot(c(2:10), bic.values, xlab="Number of clusters (k)", ylab="BIC scores", pch=1)
```




```
data.frame(k=c(2:10), bic=bic.values)
```

```
##    k      bic
## 1  2 81903.33
```

```
## 2  3 40594.58
## 3  4 31039.93
## 4  5 19555.28
## 5  6 19924.45
## 6  7 20500.80
## 7  8 21151.27
## 8  9 21711.50
## 9 10 22417.17
```

We see that the BIC score is minimum for $\mathbf{k = 5}$. Thus, this is the optimal number of clusters for our example.

Visualizing our output

```
d <- dist(t(e)) # distance between sample points
km <- kmeans(t(e), centers=5)
mds <- cmdscale(d)

mypar(1,1)
plot(mds[,1], mds[,2], col=km$cluster, pch=16)
legend(-16.2, -2.5, legend=c(1:5), col=c(1:5), pch=16)
```