HW_4.R

Shreyas

Wed Nov 29 23:45:09 2017

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.4.2
```

```
library(curl)
```

```
## Warning: package 'curl' was built under R version 3.4.2
```

```
library(cluster)
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 3.4.2
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.g
l/13EFCZ
```

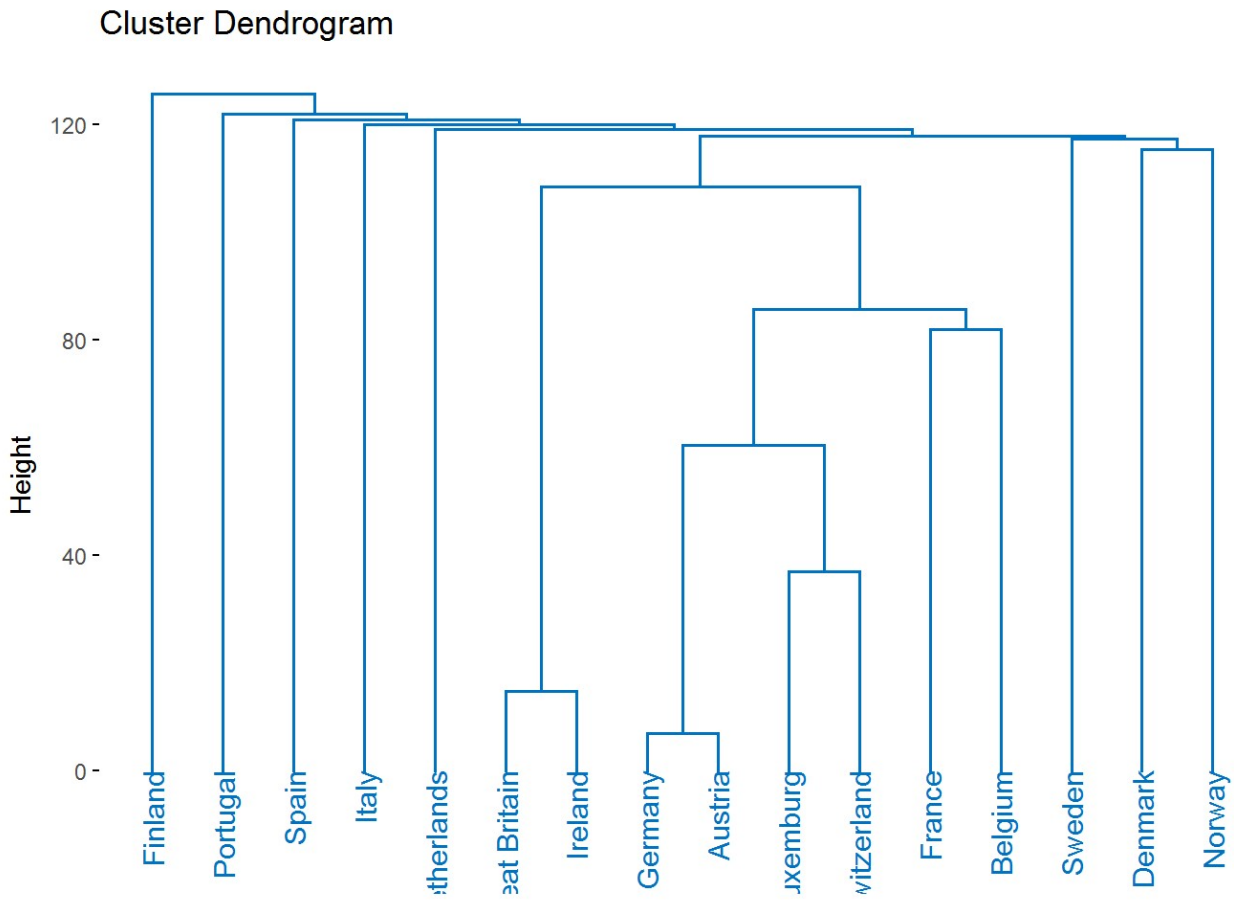
```
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.4.2
```

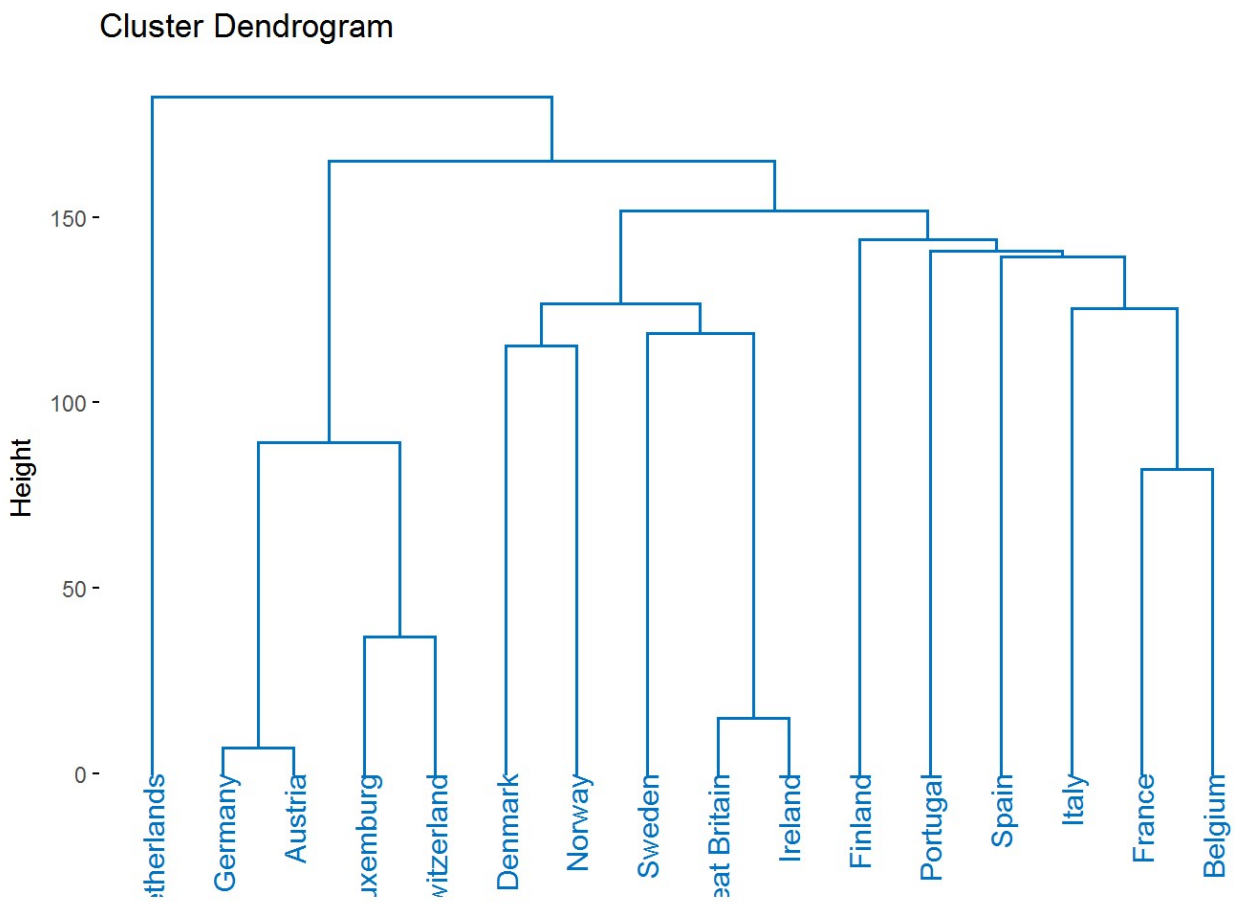
```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

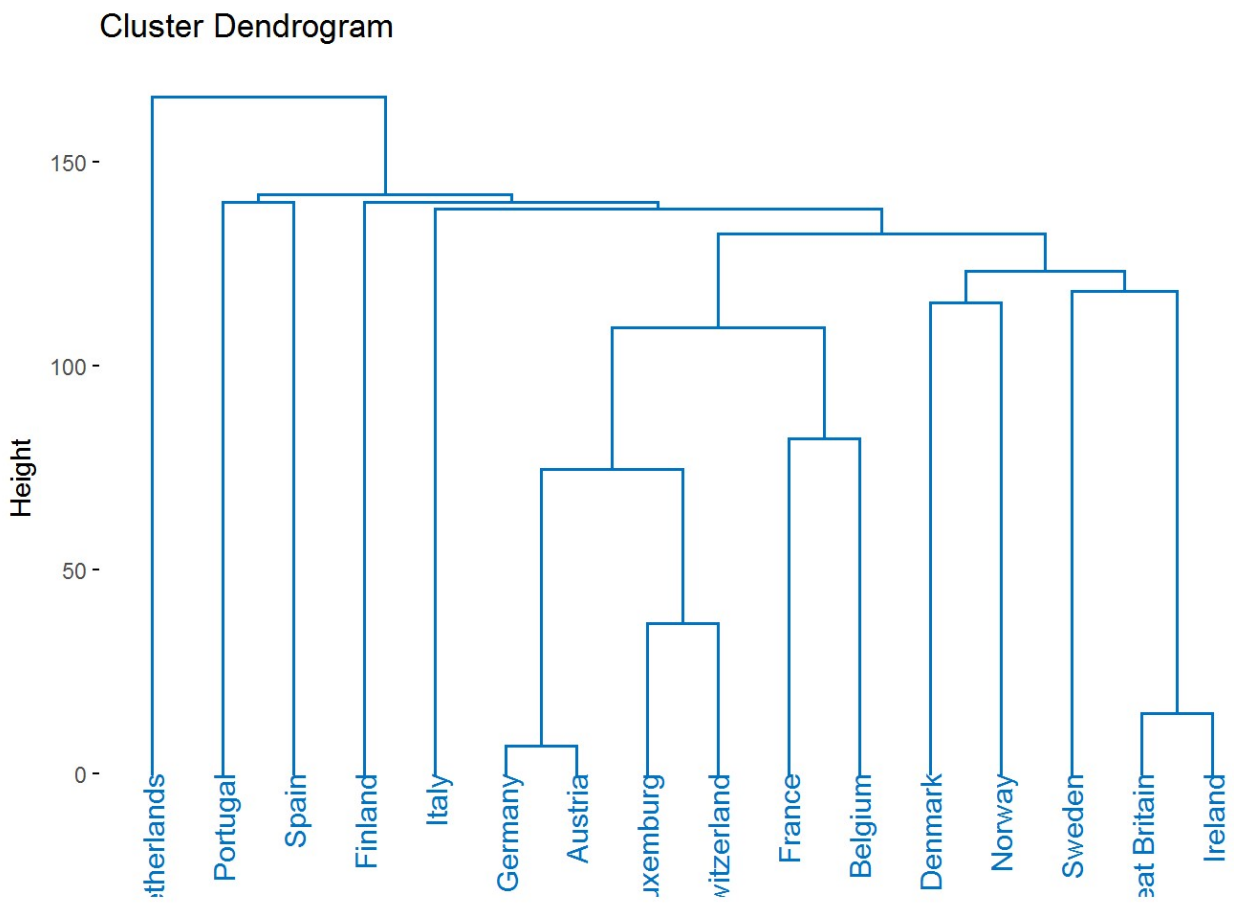
```
setwd("C:/Users/Shreyas/Desktop/DM")
rm(list=ls())

lg <- fread('https://people.sc.fsu.edu/~jburkardt/datasets/hartigan/file46.txt')
#Used to fetch the file from the address

#write.csv(lg, file = "file.csv")

lg1 <- read.csv("file.csv", header = T, sep = ",", row.names = 'Country')
lg1$X=NULL

#2.1(a)
hc.single <- factoextra::eclust(lg1, "hclust", hc_method="single")
fviz_dend(hc.single, show_labels=TRUE, palette="jco", as.ggplot=T)
```




```
hc.complete <- factoextra::eclust(lg1, "hclust", hc_method="complete")
fviz_dend(hc.complete, show_labels=TRUE, palette="jco", as.ggplot=T)
```



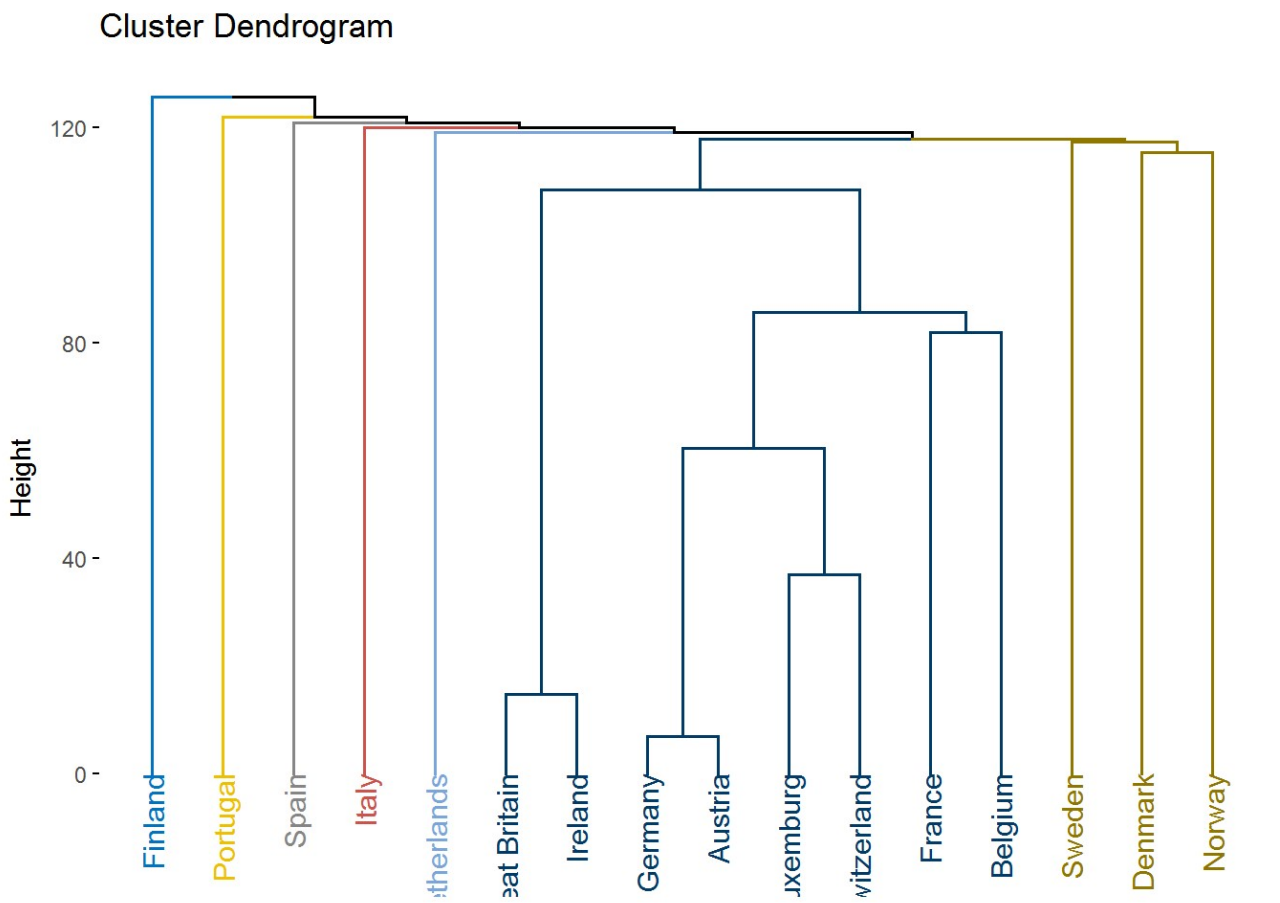

```
hc.average <- factoextra::eclust(lg1, "hclust", hc_method="average")
fviz_dend(hc.average, show_labels=TRUE, palette="jco", as.ggplot=T)
```


Cluster Dendrogram
Height
150
100
50
0
etherlands
Portugal
Spain
Finland
Italy
Germany
Austria
uxemburg
witzerland
France
Belgium
Denmark
Norway
Sweden
eat Britain
Ireland

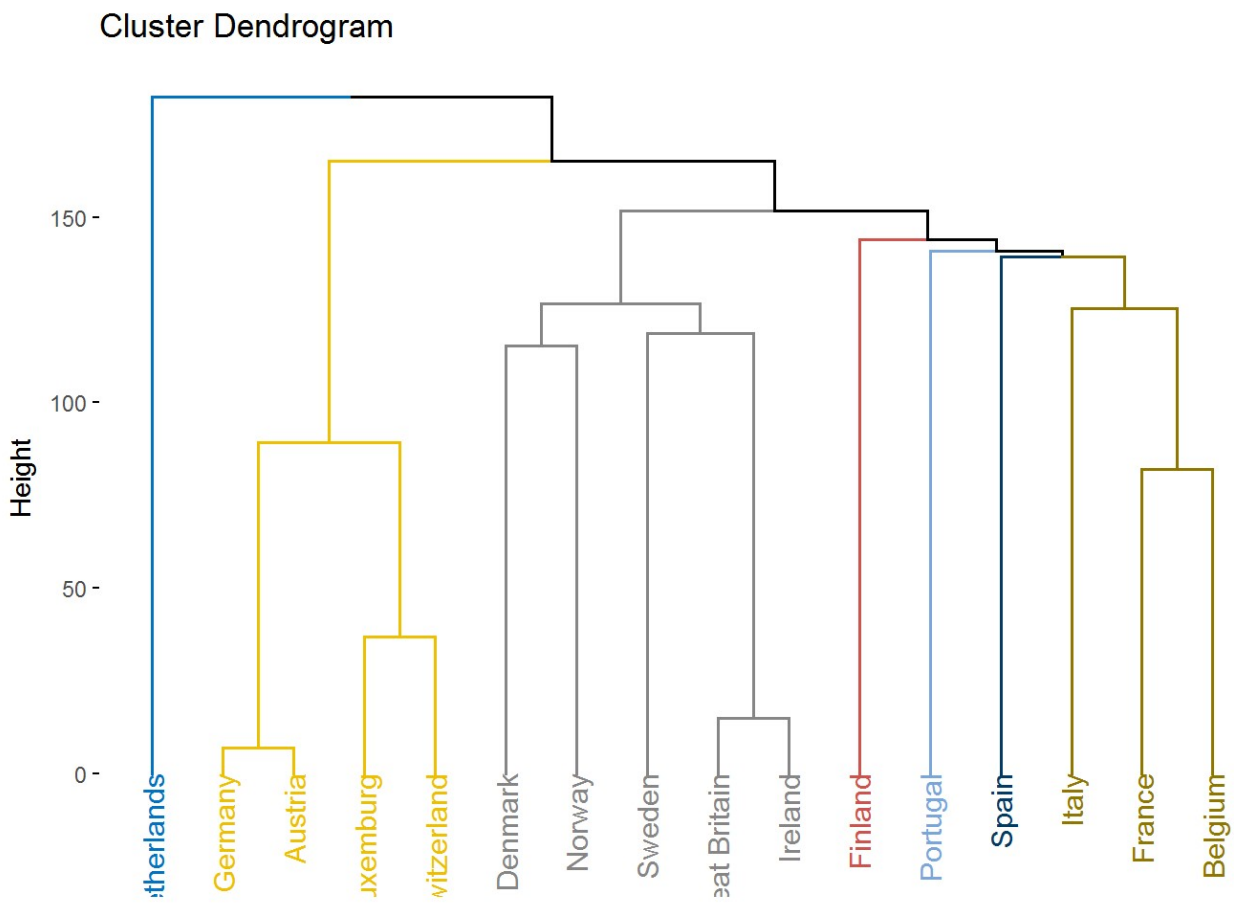
```
#2.1(b)

#1.Method : Single
#Two singleton clusters are {Great Britain,Ireland},{West Germany,Austria},{Luxemburg,
Switzerland},{France, Belgium},{Denmark, Norway}

#2.Method : Complete
# Two singleton clusters are {Denmark, Norway}, {Great Britain,Ireland},{West Germany,
Austria},{Luxemburg,Switzerland}, {France, Belgium}

#3.Method : Average
# Two singleton clusters are {Portugal,Spain},{Denmark, Norway},{France, Belgium}, {Gr
eat Britain,Ireland},{West Germany,Austria},{Luxemburg,Switzerland}

#2.1 (c)
# Italy should be clustered with a large cluster. By Looking at the raw data, Italy ha
s higher dissimilarity with other countries.
#beacuse of which, if we cut at certain cutoff, it will make Italy  a outlier and wil
l form a cluster with less dissimilarity comparitively.

#2.1 (d)
#Purity as the linkage strategy that produces the most two-singleton cluster, there i
s only one method i.e "Average".
#Linkage with method="Average" is pure by definition

#2.1 (e)
cuttree.125<-cutree(hc.average,h=125)
table(cuttree.125)
```

```
## cuttree.125
## 1 2 3 4 5 6 7
## 6 1 1 5 1 1 1
```

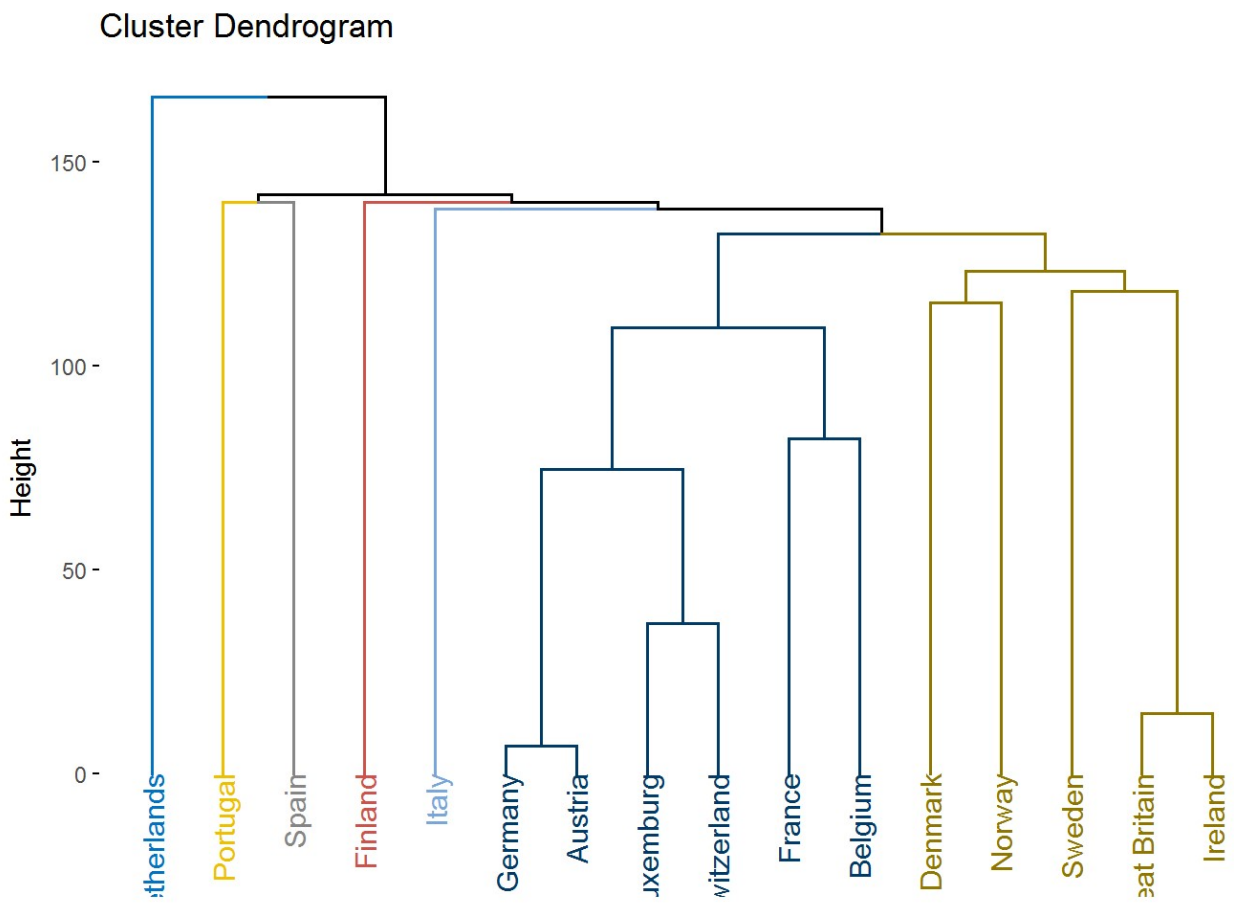
```
#There are 7 clusters at height 125.

#2.1 (f)
hc.single1 <- factoextra::eclust(lg1, "hclust", k=7, hc_method="single")
fviz_dend(hc.single1, show_labels=TRUE, palette="jco", as.ggplot=T)
```




```
hc.complete1<- factoextra::eclust(lg1, "hclust", k=7, hc_method="complete")
fviz_dend(hc.complete1, show_labels=TRUE, palette="jco", as.ggplot=T)
```




```
hc.average1 <- factoextra::eclust(lg1, "hclust",k=7, hc_method="average")
fviz_dend(hc.average1, show_labels=TRUE, palette="jco", as.ggplot=T)
```




```
#2.1(g)
library(fpc)
```

```
## Warning: package 'fpc' was built under R version 3.4.2
```

```
ct<-dist(lg1)

st <- cluster.stats(ct, hc.single1$cluster, silhouette=TRUE)
st$dunn
```

```
## [1] 0.7813006
```

```
st$avg.silwidth
```

```
## [1] 0.1215148
```

```
st1 <- cluster.stats(ct, hc.complete1$cluster, silhouette=TRUE)
st1$dunn
```

```
## [1] 0.6768822
```

```
st1$avg.silwidth
```

```
## [1] 0.1922308
```

```
st2<- cluster.stats(ct, hc.average1$cluster,silhouette=TRUE)
st2$dunn
```

```
## [1] 0.807345
```

```
st2$avg.silwidth
```

```
## [1] 0.1698248
```

```
#2.1(h)
#Since, the Dunn index for hc.average is maximum. Thus, hc.average is the best cluste
r.

#2.1(i)
#Since, silhouette width for hc.complete is maximum. Thus, hc.complete is the best clu
ster.



#2.2
library(textreuse)
```

```
## Warning: package 'textreuse' was built under R version 3.4.2
```

```
files <- list.files("C:/Users/Shreyas/Desktop/corpus/corpus", full.names = T)
minhash <- minhash_generator(n=160, seed=100)

#2.2(a)
corpus <- TextReuseCorpus(files, tokenizer = tokenize_ngrams, n = 5,
                          minhash_func = minhash, keep_tokens = TRUE)
length(unlist(tokens(corpus)))
```

```
## [1] 22075
```

```
#2.2(b)
library(magrittr)
tot_tokens <- tokens(corpus)
corpusMat <- list.files("C:/Users/Shreyas/Desktop/corpus/corpus", full.names=F)
doc_dict <- unlist(tokens(corpus)) %>% unique()
Matr <- lapply(tot_tokens, function(set, dict) {   as.integer(dict %in% set)}, dict =
doc_dict) %>% data.frame()
tempSetName <-setNames( Matr, paste( corpusMat, 1:length(corpusMat)) )
rownames(Matr) <- doc_dict
dim(Matr)
```

```
## [1] 17614   100
```

```
#i.e. 17614*100

#2.2(c)
tokens(corpus[["orig_taske"]])[1:5]
```

```
## [1] "in mathematics and computer science"
## [2] "mathematics and computer science dynamic"
## [3] "and computer science dynamic programming"
## [4] "computer science dynamic programming is"
## [5] "science dynamic programming is a"
```

```
#2.2(d)
#As we choose only 240 rows for signature matrix, The dimensions of signature matrix i
s 240*100.
# and we have generated Characteristic matrix of dimension 17614*100.
# therefore, the reduction of size of problem is 98.637%.

#2.2(e)
lsh_probability(h = 240, b =  60, s = 0.3)
```

```
## [1] 0.3861342
```

```
#This probability is less than the required one and thus increase the bands to 80.
lsh_probability(h = 240, b =  80, s = 0.3)
```

```
## [1] 0.8880492
```

```
#We get the required probability i.e 80 at bands=80.

#2.2(f)
buckets <- lsh(corpus, bands = 80)
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.2
```

```
candidates <- lsh_candidates(buckets)
ncandidatepair<-nrow(candidates)
#Number of candidate pairs
ncandidatepair
```

```
## [1] 72
```

```
#2.2(g)
lsh_Result <- lsh_compare(candidates, corpus, jaccard_similarity)
lsh_Result[order(lsh_Result$score,decreasing = TRUE),][1:5,]
```

```
## # A tibble: 5 x 3
##            a          b     score
##        <chr>      <chr>     <dbl>
## 1 g4pC_taska orig_taska 0.7896254
## 2 g3pA_taskd orig_taskd 0.7186630
## 3 g4pB_taske orig_taske 0.4491803
## 4 g3pA_taskd g4pC_taskd 0.4142857
## 5 g2pB_taske orig_taske 0.3982906
```

```
#2.2(h)
# If we don't use the Locality Sensative Hashing and directly examine every pair for s
imilarity then
# The Number of pairs of documents to be examined will be = (No of Documents)C2
# Here we can write as No of Documents = 100
# No of pairs = 100C2 = 100!/(98!*2!) = 4950

# Solution for 2.2 (h)(ii)
# No of candidate pairs generated in 2.2 (f) = 72.
# The ratio of doc pair to candidate pair number is : 4950/72 = 68.75
# It shows that if we dont do Locality Sensative Hashing the number of comparisons we
have to do is 68.75 times than number of comparisons we will do after doing Locality S
ensative Hashing.

#2.3(a)

item<-read.csv("C:/Users/Shreyas/Desktop/DM/ml-100k/u.item", sep = "|",comment.char =
"#")
item$X1=NULL
data<-read.csv("C:/Users/Shreyas/Desktop/DM/ml-100k/u.data", sep = "\t", header = T,co
mment.char = "#")
#2.3(i)
user200_rownumber <- which(data$user== 200)
user50_rownumber <- which(data$user == 50)

user200 <- data[user200_rownumber,]
user50 <- data[user50_rownumber,]

movies200 <-item[user200[,2],]
movies50 <- item[user50[,2],]

movie.matrix200 <- movies200[,6:24]
genre200 <- apply(movie.matrix200,2,mean)
vector200 <- as.vector(genre200)

movie.matrix50 <- movies50[,6:24]
genre50 <- apply(movie.matrix50,2,mean)
vector50 <- as.vector(genre50)

cosine <- function(x, y) {
  # Need to do error checking:
  # 1) Ensure x and y are vectors.

  sum(x*y)/(norm(x, type="2") * norm(y, type="2"))
}

cosine(vector50,vector200)
```

```
## [1] 0.54825
```

```
#cluster_similarity(vector200, vector50, similarity="jaccard", method="independence")

#2.3(ii)
movie127 <- item[127,]
vector127 <- as.vector(movie127[,6:24])
cosine(vector127,vector50)
```

```
## [1] 0.6235022
```

```
#2.3(iii)
cosine(vector127,vector200)
```

```
## [1] 0.5533398
```

```
#2.3(iv)
#the movie 127 will be recommended to which user 50.


#2.3(b)

utilitymatrix<-matrix(0,6,11)

for(i in 1:length(unlist(data[,1])))
{
  if(data[i,1]==1 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],1]<-data[i,3]
  }

  if(data[i,1]==21 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],2]<-data[i,3]
  }
  if(data[i,1]==44 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],3]<-data[i,3]
  }
  if(data[i,1]==59 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],4]<-data[i,3]
  }
  if(data[i,1]==72 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],5]<-data[i,3]
  }
  if(data[i,1]==82 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],6]<-data[i,3]
  }
  if(data[i,1]==102 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],7]<-data[i,3]
  }
  if(data[i,1]==234 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],8]<-data[i,3]
  }
  if(data[i,1]==268 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],9]<-data[i,3]
  }
  if(data[i,1]==409 && data[i,2]<7)
```

```
  {
    utilitymatrix[data[i,2],10]<-data[i,3]
  }
  if(data[i,1]==486 && data[i,2]<7)
  {
    utilitymatrix[data[i,2],11]<-data[i,3]
  }
}
colnames(utilitymatrix)<-c("user1","user21","user44","user59","user72","user82","user1
02","user234","user268","user409","user486")

#View(utilitymatrix)


means <- apply(utilitymatrix, 1, function(x) mean(x, na.rm=T))

means
```

```
## [1] 3.363636 1.090909 1.181818 1.545455 1.727273 1.181818
```

```
for (i in 1:dim(utilitymatrix)[1]) {
  for (j in 1:dim(utilitymatrix)[2])
  {
    if(utilitymatrix[i,j]>0)
    {
      utilitymatrix[i,j] <- utilitymatrix[i,j] - means[i]
    }
  }
}
similarmovie<-matrix(0,6,1)

for (i in 1:dim(utilitymatrix)[1])
{
  similarmovie[i,1]<-round(cosine(utilitymatrix[5,], utilitymatrix[i, ]), digits=2)
}


rownames(similarmovie)<-c("1","2","3","4","5","6")
a<-as.numeric(rownames(similarmovie)[order(similarmovie, decreasing=TRUE)][1:6])

r<-((similarmovie[a[2],1]*utilitymatrix[a[2],9])+(similarmovie[a[3],1]*utilitymatrix[a
[3],9])+(similarmovie[a[4],1]*utilitymatrix[a[4],9]))/(similarmovie[a[2],1]+similarmov
ie[a[3],1]+similarmovie[a[4],1])
r
```

```
##         2
## 0.9059123
```