ashishsa_hw5_p2

```
library(glasso)
library(bestNormalize)
```

```
## Warning: package 'bestNormalize' was built under R version 3.6.3
```

```
library(cluster)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
library(ggdendro)
```

```
## Warning: package 'ggdendro' was built under R version 3.6.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.6.3
```

```
##
## 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(dendextend)
```

```
## Warning: package 'dendextend' was built under R version 3.6.3
```

```
##
## ---------------------
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following object is masked from 'package:ggdendro':
##
##     theme_dendro
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.3
```

```
library(cluster)
library(RColorBrewer)
library(ape)
```

```
## Warning: package 'ape' was built under R version 3.6.3
```

```
##
## Attaching package: 'ape'
```

```
## The following objects are masked from 'package:dendextend':
##
##     ladderize, rotate
```

```
library(fossil)
```

```
## Warning: package 'fossil' was built under R version 3.6.3
```

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

```
## Warning: package 'sp' was built under R version 3.6.3
```

```
## Loading required package: maps
```

```
## Warning: package 'maps' was built under R version 3.6.3
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:cluster':
##
##     votes.repub
```

```
## Loading required package: shapefiles
```

```
## Warning: package 'shapefiles' was built under R version 3.6.3
```

```
## Loading required package: foreign
```

```
##
## Attaching package: 'shapefiles'
```

```
## The following objects are masked from 'package:foreign':
##
##     read.dbf, write.dbf
```

```
library(kohonen)
```

```
## Warning: package 'kohonen' was built under R version 3.6.3
```

```
##
## Attaching package: 'kohonen'
```

```
## The following object is masked from 'package:maps':
##
##     map
```

```
library(glasso)
library(gRain)
```

```
## Warning: package 'gRain' was built under R version 3.6.3
```

```
## Loading required package: gRbase
```

```
## Warning: package 'gRbase' was built under R version 3.6.3
```

```
## Required packages from Bioconductor are not installed: RBGL
## Please execute these lines and re-install gRbase again:
## source("https://bioconductor.org/biocLite.R");biocLite(c("graph", "RBGL", "Rgraphviz"))
```

```
library(graph)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:dplyr':
##
##     combine, intersect, setdiff, union
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min
```

```
##
## Attaching package: 'graph'
```

```
## The following objects are masked from 'package:ape':
##
##     complement, edges
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

```
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 3.6.3
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
##
## Attaching package: 'GGally'
## The following object is masked from 'package:dplyr':
##
##     nasa
library(blob)
## Warning: package 'blob' was built under R version 3.6.3
library(AnnotationDbi)
## Loading required package: stats4
## Loading required package: Biobase
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
##
## Attaching package: 'Biobase'
## The following object is masked from 'package:gRbase':
##
##     description<-
## Loading required package: IRanges
## Loading required package: S4Vectors
## Warning: package 'S4Vectors' was built under R version 3.6.3
##
## Attaching package: 'S4Vectors'
## The following objects are masked from 'package:dplyr':
##
##     first, rename
## The following object is masked from 'package:base':
##
##     expand.grid
##
## Attaching package: 'IRanges'
## The following object is masked from 'package:sp':
##
##     %over%
## The following objects are masked from 'package:dplyr':
##
##     collapse, desc, slice
```

```
## The following object is masked from 'package:grDevices':
##
##     windows
```

```
##
## Attaching package: 'AnnotationDbi'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(geneplotter)
```

```
## Loading required package: lattice
```

```
## Loading required package: annotate
```

```
## Loading required package: XML
```

```
## Warning: package 'XML' was built under R version 3.6.3
```

```
##
## Attaching package: 'XML'
```

```
## The following object is masked from 'package:graph':
##
##     addNode
```

```
##
## Attaching package: 'annotate'
```

```
## The following object is masked from 'package:gRain':
##
##     getEvidence
```

```
data("state.x77")
```

```
## Warning in data("state.x77"): data set 'state.x77' not found
```

```
df1 <- state.x77
head(df1)
```

```
##            Population Income Illiteracy Life Exp Murder HS Grad Frost   Area
## Alabama          3615   3624        2.1    69.05   15.1    41.3    20  50708
## Alaska            365   6315        1.5    69.31   11.3    66.7   152 566432
## Arizona          2212   4530        1.8    70.55    7.8    58.1    15 113417
## Arkansas         2110   3378        1.9    70.66   10.1    39.9    65  51945
## California      21198   5114        1.1    71.71   10.3    62.6    20 156361
## Colorado         2541   4884        0.7    72.06    6.8    63.9   166 103766
```

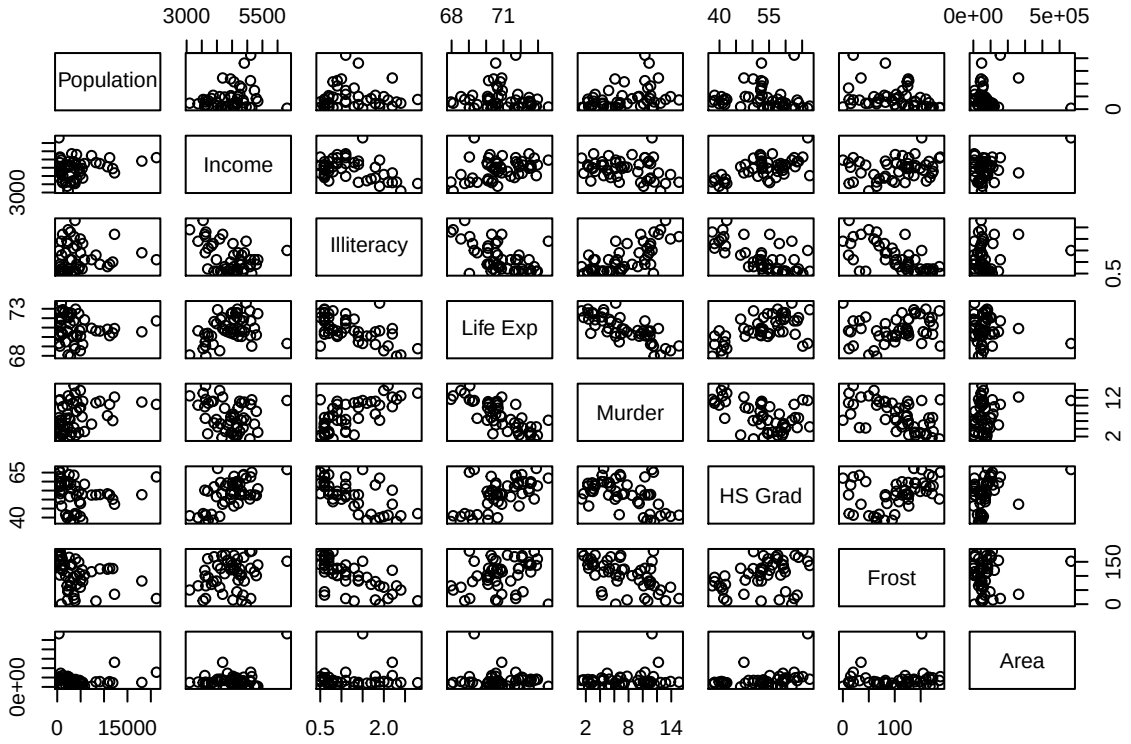
We are now going to cluster the states by performing Heirarchical Clustering with complete linkage and eucledian distance to cluster the states.

```
nrow(df1)
```

```
## [1] 50
```

We observe that there are 8 columns(Population, Income, Illiteracy, Life Exp, Murder, HS Grad, Frost, Area). There are 50 Rows here.

```
pairs(df1)
```



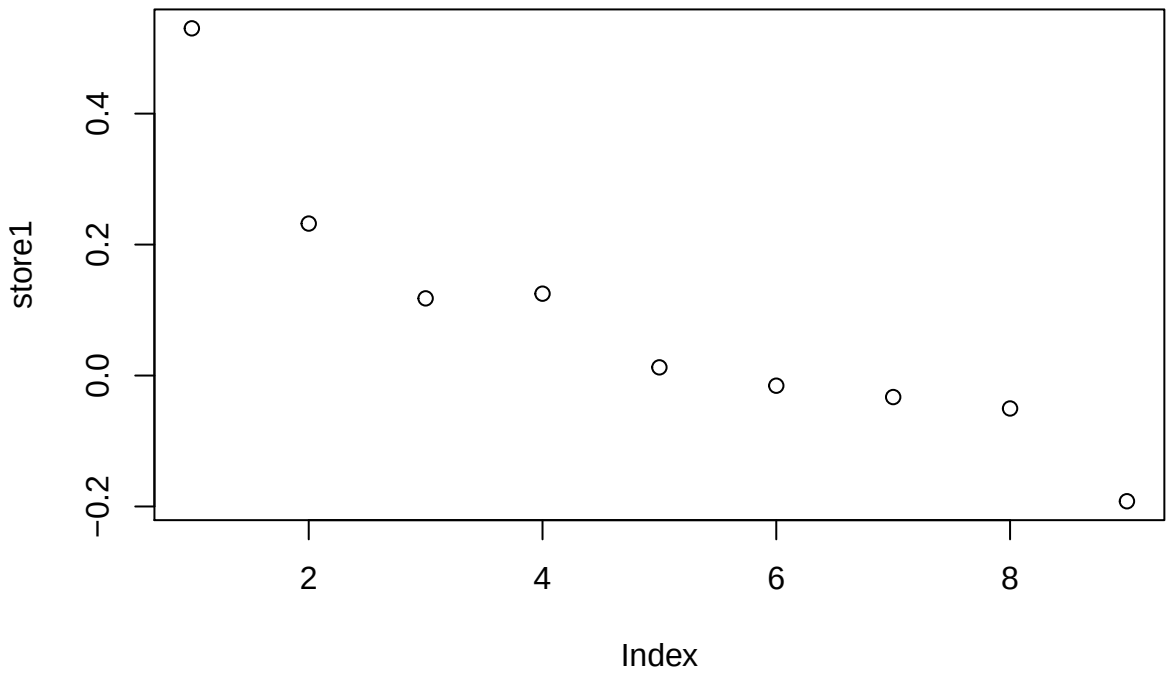
We now scale the data

```
m <- apply(df1,2,mean)
s <- apply(df1,2,sd)
z <- scale(df1,m,s)
```

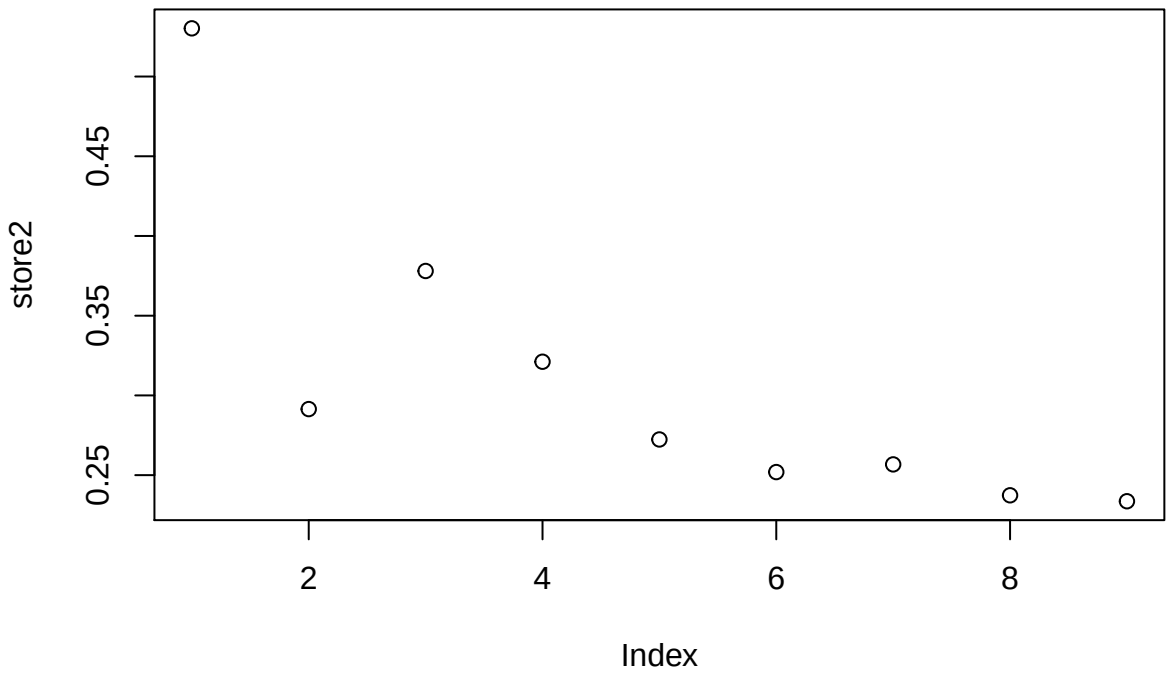
We compute the Eucledian Distance between the data points and calculate the clusters using single,average and complete clusters.

##ERROR RATES FOR VARIOUS METHODS OF HEIRARCHICAL CLUSTERING The plot that has the lowest error for the best k-value gives us the best result.

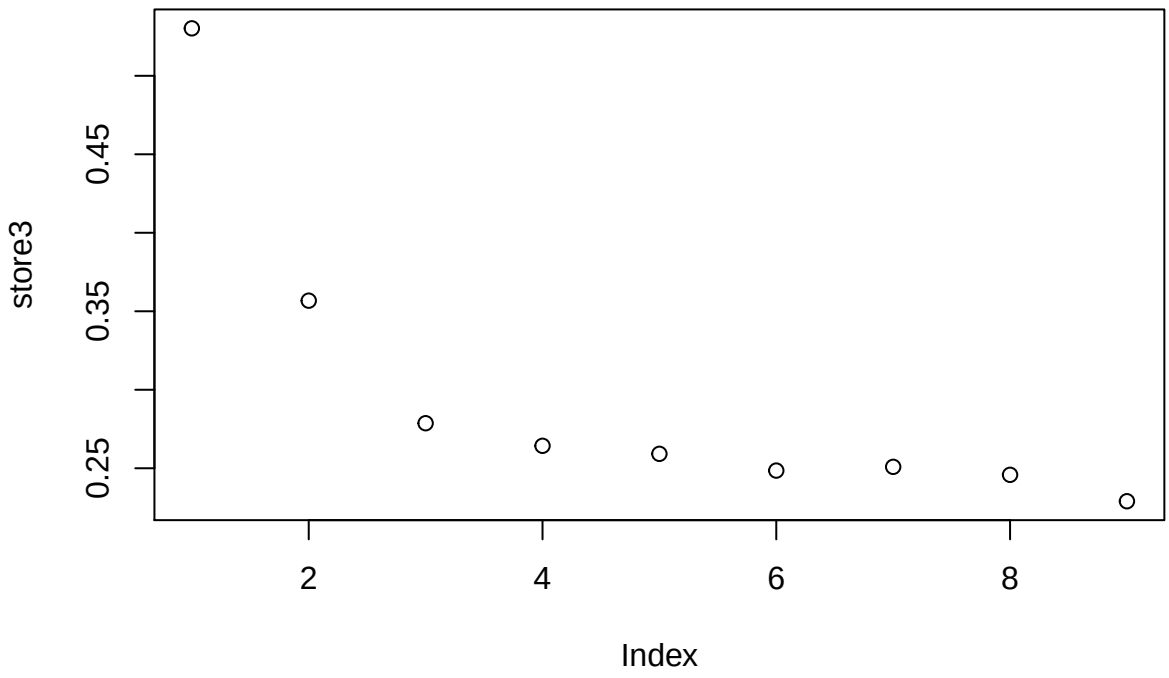
```
store1 <- c()
for (i in 2:10){
  ct <- cutree(hclust1,k=i)
  si <- silhouette(ct,dist = dist1)
  avg_width <- summary(si)$avg.width
  store1 <- c(store1,avg_width)
}
plot(store1)
```




```
store2 <- c()
for (i in 2:10){
  ct <- cutree(hclust2,k=i)
  si <- silhouette(ct,dist = dist1)
  avg_width <- summary(si)$avg.width
  store2 <- c(store2,avg_width)
}
plot(store2)
```




```
library(ggplot2)
store3 <- c()
for (i in 2:10){
  ct <- cutree(hclust3,k=i)
  si <- silhouette(ct,dist = dist1)
  avg_width <- summary(si)$avg.width
  store3 <- c(store3,avg_width)
}

plot(store3)
```



From the above plots we can estimate that

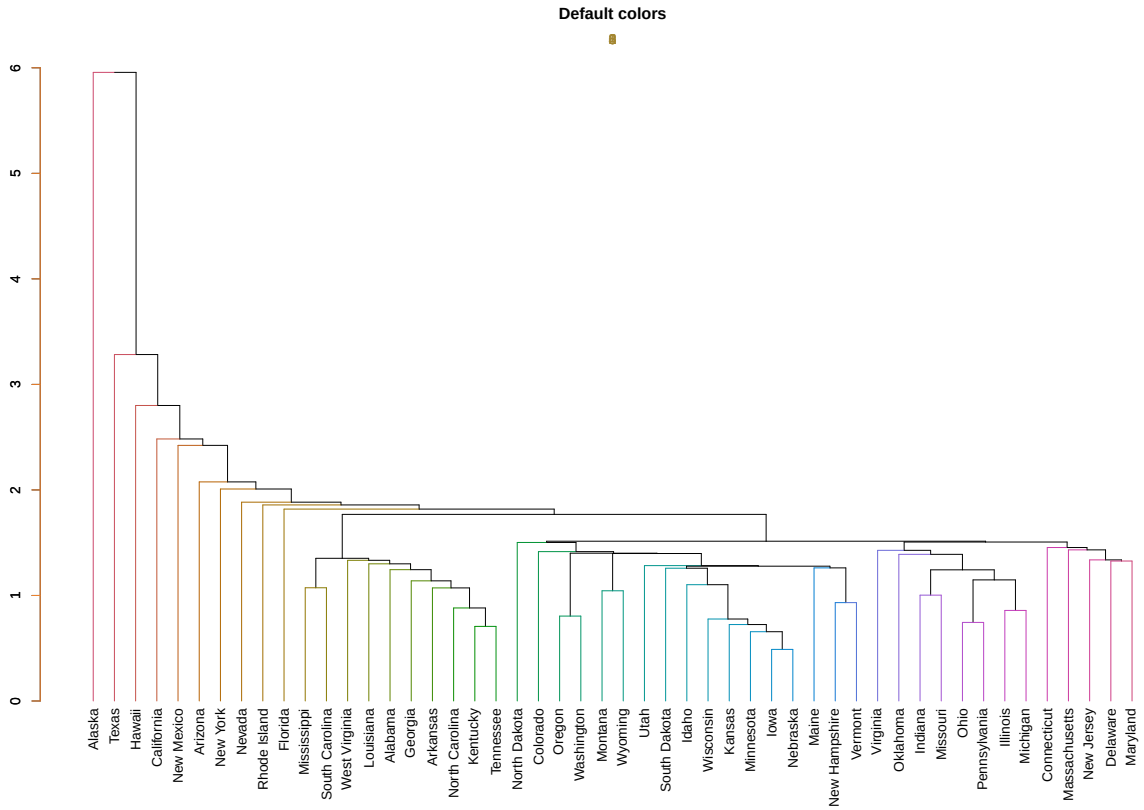
1)single linkage has the error of 0.17 for k=4(which is estimated to be best cluster size)

2)average linkage has the error of 0.33 for k=4(which is estimated to be best cluster size)

3)complete linkage has the error of 0.28 for k=4(which is estimated to be best cluster size)

We cluster the data using Single Linkage

```
dend1 <- as.dendrogram(hclust1)
dend1 %>% set("branches_k_color") %>%
  plot(main = "Default colors") %>%
  axis(side = 2,col = "#F38630",labels = TRUE) %>%
  mtext(col = "#A38630")
```

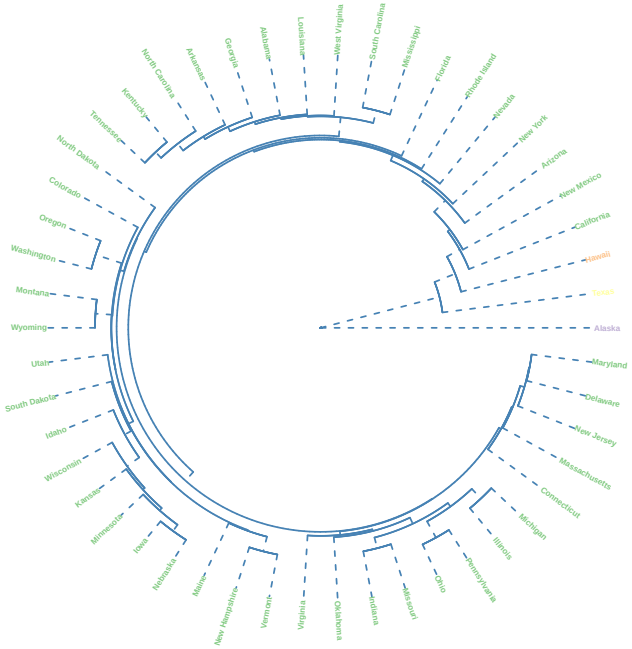


Incase of Single Linkage of Heirarchical Clustering we observe that there are a large number of clusters. We find the Minimal Inter-Cluster Similarity which is used to calculate the inter-cluster similarity. We now plot the clusters using Fan-Plot.

```
cutree1 <- cutree(dend1,4)
cutree1
```

```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              2              1              1              1
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              1              1              1              1              1
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              3              1              1              1              1
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              1              1              1              1              1
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              1              1              1              1              1
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              1              1              1              1              1
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              1              1              1              1              1
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              1              1              1              1              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              1              1              4              1              1
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              1              1              1              1              1
```

```
plot(as.phylo(hclust1), type = "fan", cex = 0.6,
    tip.color = brewer.pal(4, 'Accent')[cutree1],
      font = 2,
     edge.color = 'steelblue', edge.width = 2, edge.lty = 2)
```

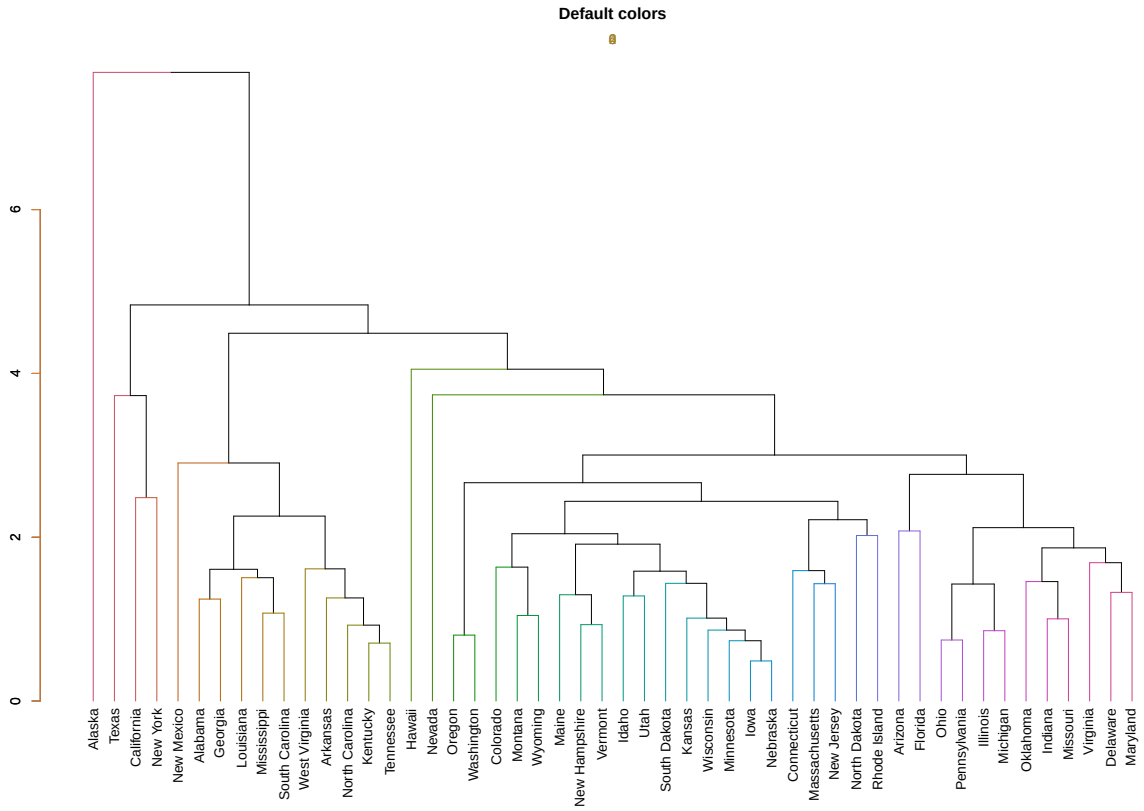


As we can observe from the fan plot most of thee data gets clustered into the same class.

##AVERAGE LINKAGE

We cluster the data using Average Linkage.

```
dend2 <- as.dendrogram(hclust2)
dend2 %>% set("branches_k_color") %>%
  plot(main = "Default colors") %>%
  axis(side = 2,col = "#F38630",labels = TRUE) %>%
  mtext(col = "#A38630")
```



We compute Mean intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the average of these dissimilarities.

```
cutree2 <- cutree(dend2,4)
cutree2
```

```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              2              3              1              4
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              3              3              3              3              1
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              3              3              3              3              3
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              3              1              1              3              3
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              3              3              3              1              3
##        Montana       Nebraska         Nevada New Hampshire     New Jersey
##              3              3              3              3              3
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              1              4              1              3              3
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              3              3              3              3              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              3              1              4              3              3
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              3              3              1              3              3
```

```
plot(as.phylo(hclust2), type = "fan", cex = 0.6,
```

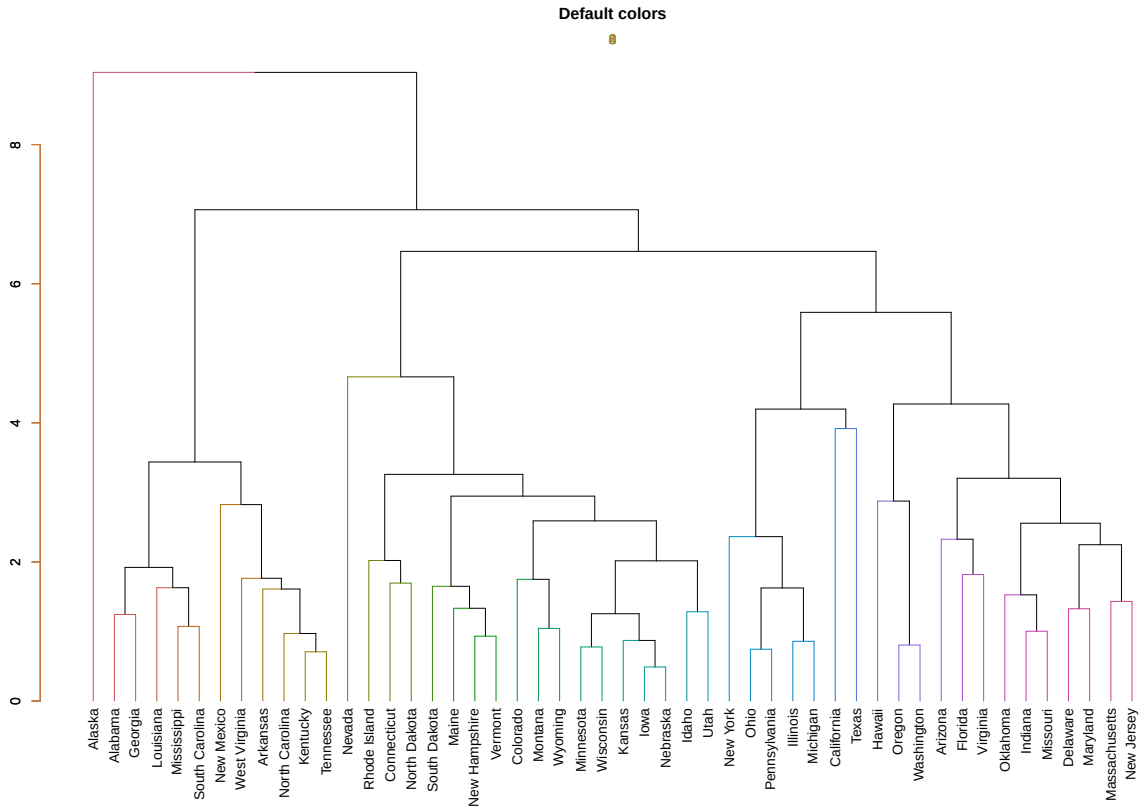
```
    tip.color = brewer.pal(4, 'Accent')[cutree2],
      font = 2,
     edge.color = 'steelblue', edge.width = 2, edge.lty = 2)
```

As we can observe from the fan plot the data gets clustered into four classes.

##COMPLETE LINKAGE

We cluster the data using Complete Linkage.

```
dend3 <- as.dendrogram(hclust3)
dend3 %>% set("branches_k_color") %>%
  plot(main = "Default colors") %>%
  axis(side = 2,col = "#F38630",labels = TRUE) %>%
  mtext(col = "#A38630")
```



We compute Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the largest of these dissimilarities

```
cutree3 <- cutree(dend3,4)
cutree3
```

```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              2              3              1              3
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              4              4              3              3              1
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              3              4              3              3              4
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              4              1              1              4              3
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              3              3              4              1              3
##        Montana       Nebraska         Nevada New Hampshire     New Jersey
##              4              4              4              4              3
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              1              3              1              4              3
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              3              3              3              4              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              4              1              3              4              4
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              3              3              1              4              4
```

As we can observe from the fan plot the data gets clustered into three classes.

```
plot(as.phylo(hclust3), type = "fan", cex = 0.6,
    tip.color = brewer.pal(4, 'Accent')[cutree3],
      font = 2,
     edge.color = 'steelblue', edge.width = 2, edge.lty = 2)
```

##RESULT

From the Results of the Above Clustering we observe that the Heirarchical Clustering using Complete Linkage gives us the best results. The Cluster using the Single Linkage gives us the worst results (The tree is completely unbalanced) and the cluster uysing the Average Linkage gives us moderate results. The tree is relatively unbalanced in comparison to the complete linkage tree.

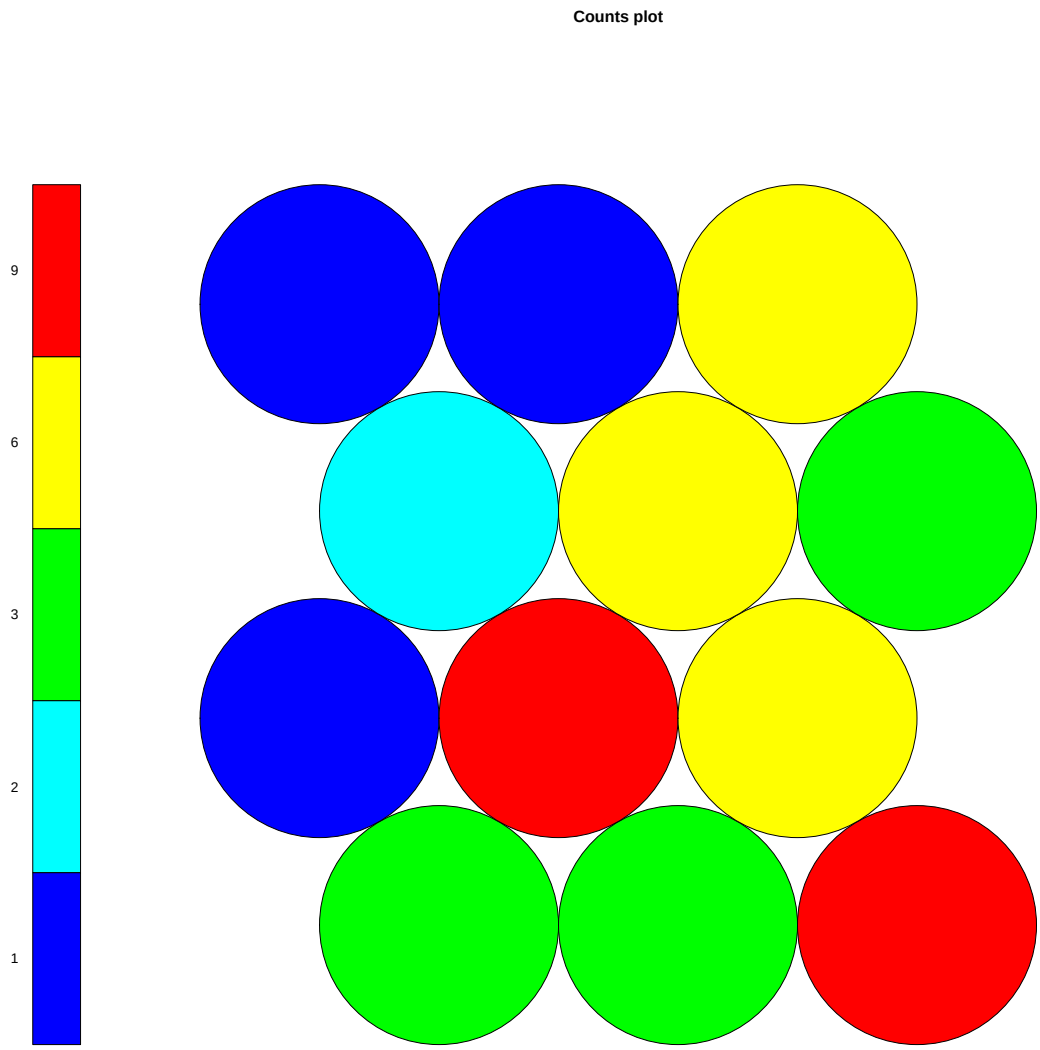
##SOM

```
training_data <- as.matrix(df1)
```

```
data_train_matrix <- training_data
som_grid <- somgrid(xdim = 3, ydim=4, topo="hexagonal")
som_model <- som(data_train_matrix,grid=som_grid,rlen=10000)
```

The counts lot gives us the count of the number of states map into the different units.

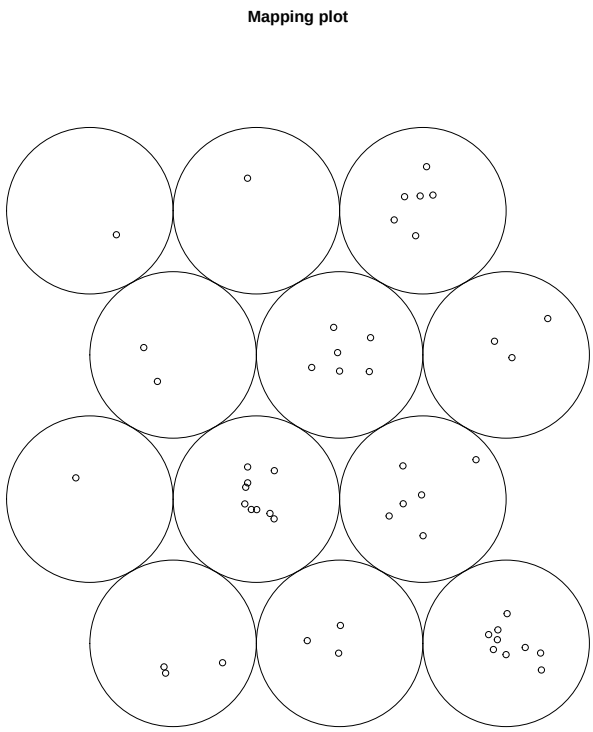
```
coolBlueHotRed <- function(n,alpha=1){rainbow(n,end=4/6,alpha=alpha)[n:1]}
plot(som_model,type="counts",palette.name=coolBlueHotRed)
```



We now plot all this Information onto a Grid. This further reinforcees the information provided by the above

Heat Map.

```
plot(som_model,type="mapping")
```



This gives us the summary of the Self Organization Map.

```
summary(som_model)
```

```
## SOM of size 3x4 with a hexagonal topology and a bubble neighbourhood function.
## The number of data layers is 1.
## Distance measure(s) used: sumofsquares.
## Training data included: 50 objects.
## Mean distance to the closest unit in the map: 25596488.
```

The codes give us the list of 16 different prototypes and the code books for all of the different prototypes.

```
som_model$codes
```

```
## [[1]]
##     Population   Income Illiteracy Life Exp    Murder  HS Grad     Frost
## V1   13134.414 4611.274  1.0365343 70.60132  7.856962 51.94289 113.42847
## V2    1845.000 3647.669  1.4260493 69.34582  6.779247 44.85080 110.17731
## V3    2762.709 4823.462  1.0323691 71.36852  4.886884 54.17885 119.44644
## V4    3559.000 4864.000  0.6000000 71.72000  4.300000 63.50000  32.00000
## V5    5683.992 4292.043  1.3208021 70.52484  9.561475 48.23444  88.46988
## V6    3899.818 3849.140  1.7646432 69.66483 10.700530 43.45811  73.74988
## V7   11438.193 4747.983  0.8613972 71.16121  7.770811 60.97750  84.42275
## V8    1939.487 4387.533  0.5840715 72.55347  3.375114 59.34190 143.64342
## V9    2594.424 4459.593  0.9018888 71.68469  5.492391 50.28691 127.03917
## V10    365.000 6315.000  1.5000000 69.31000 11.300000 66.70000 152.00000
```

```
## V11  12237.000 4188.000  2.2000000 70.90000 12.200000 47.40000  35.00000
## V12   1592.944 4522.933  1.0861539 70.86028  7.561169 60.51355 111.32892
##           Area
## V1   44315.916
## V2   28103.477
## V3    6106.786
## V4   66570.000
## V5   53970.986
## V6   41558.921
## V7  151219.588
## V8   79588.362
## V9   69021.989
## V10 566432.000
## V11 262134.000
## V12 106748.034
```

We now analyze the list of all the states and the prototypes that are closest to them.

```
som_model$unit.classif
```

```
##  [1]  5 10 12  5  7 12  3  3  5  5  3  8  5  6  5  8  6  6  2  3  3  5  8  6  9
## [26]  7  8 12  3  3 12  1  5  9  1  9 12  1  3  2  8  6 11  8  3  6  4  2  5 12
```

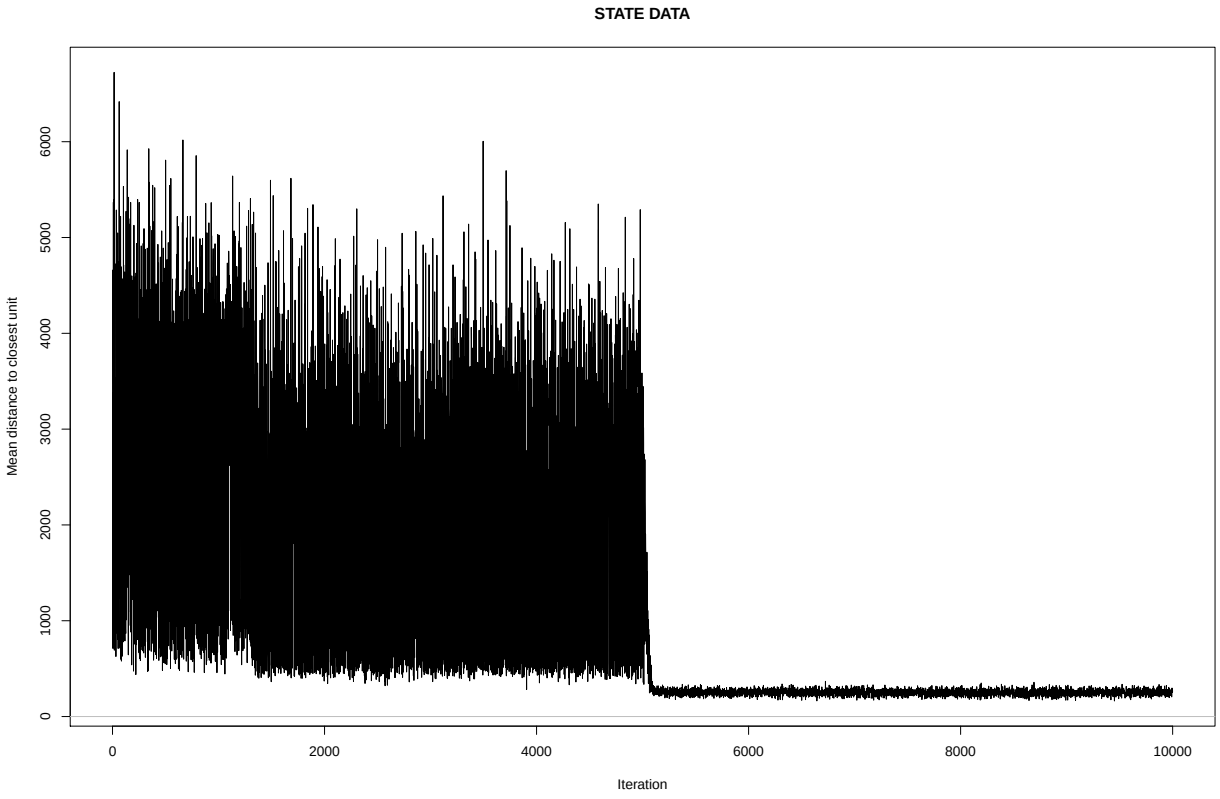
We now analyze the relative changes of the prototype code book vectors as we proceed through 200 different iteration.

```
head(som_model$changes)
```

```
##           [,1]
## [1,] 3750.1687
## [2,] 4659.9584
## [3,] 2979.3692
## [4,]  710.8409
## [5,] 1344.6151
## [6,] 3378.7028
```

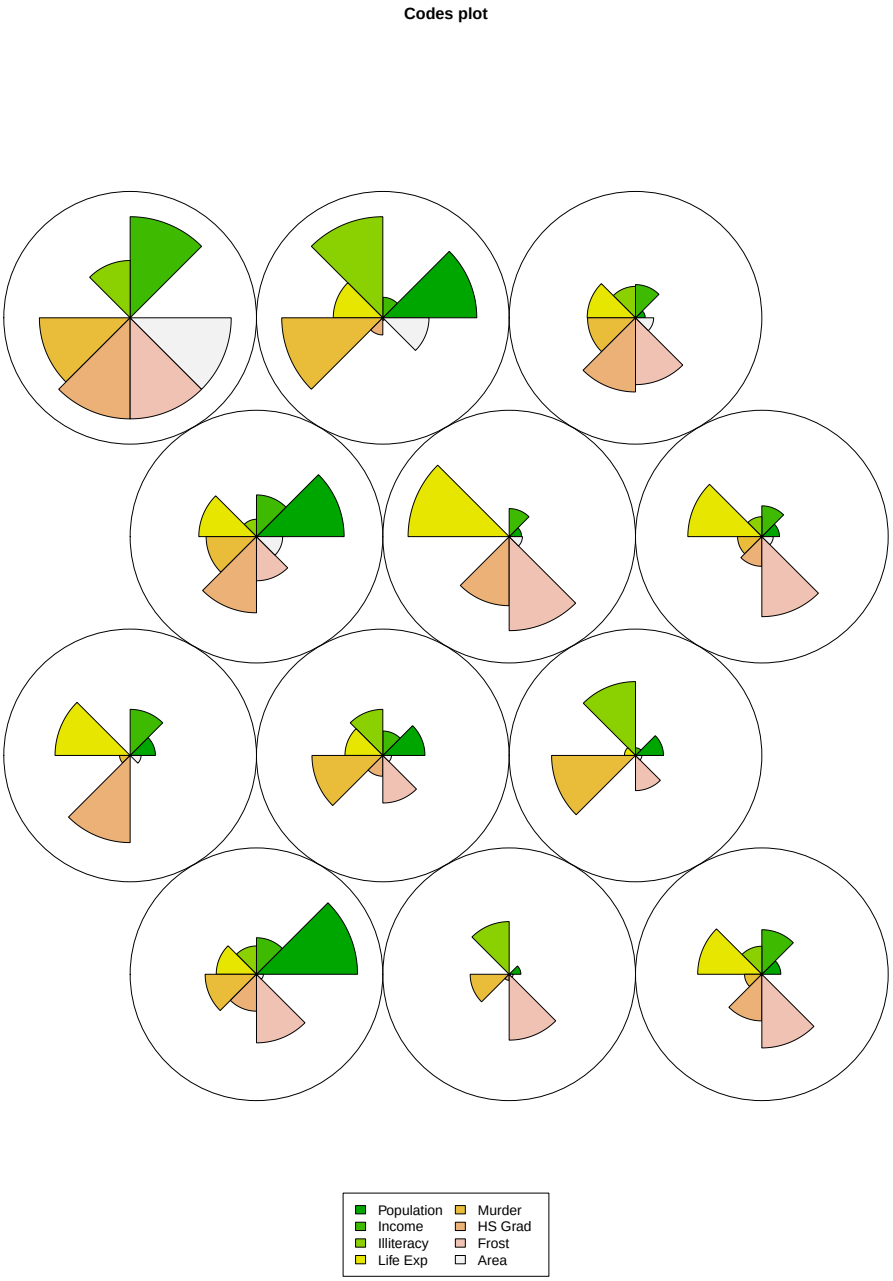
We observe that the changes start decreasing as we start increasing the number of iterations. A better understanding can be obtained from the Graph which represents the changes in the prototype. The things start converging as we move in the iteration.

```
plot(som_model,type="changes",main="STATE DATA")
```



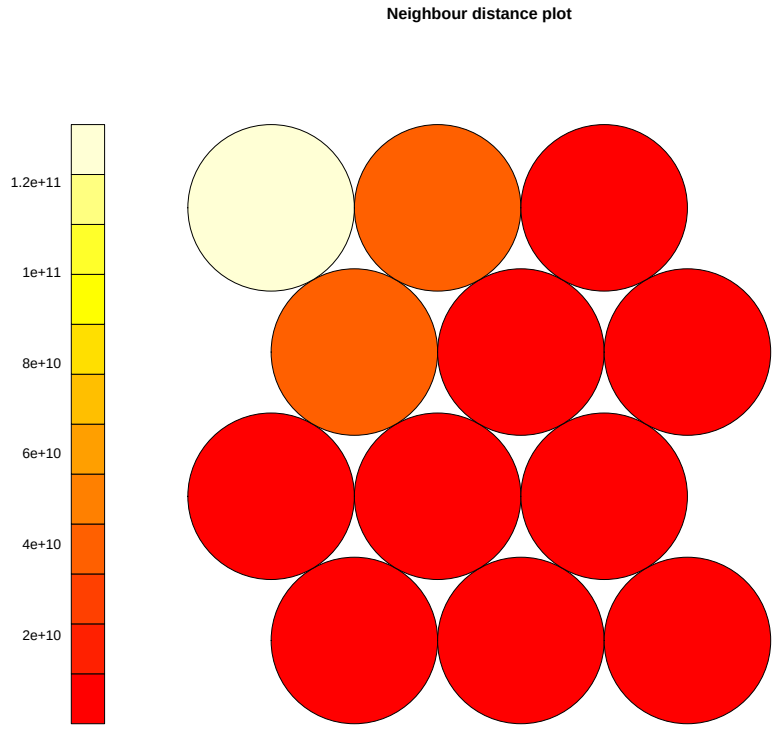
We now plot the Codes plot which gives us the distribution of various crimes across various nodes.

```
plot(som_model,type = "codes")
```



We now plot the distance to the Neighbours between each states that are plotted into a particular node.

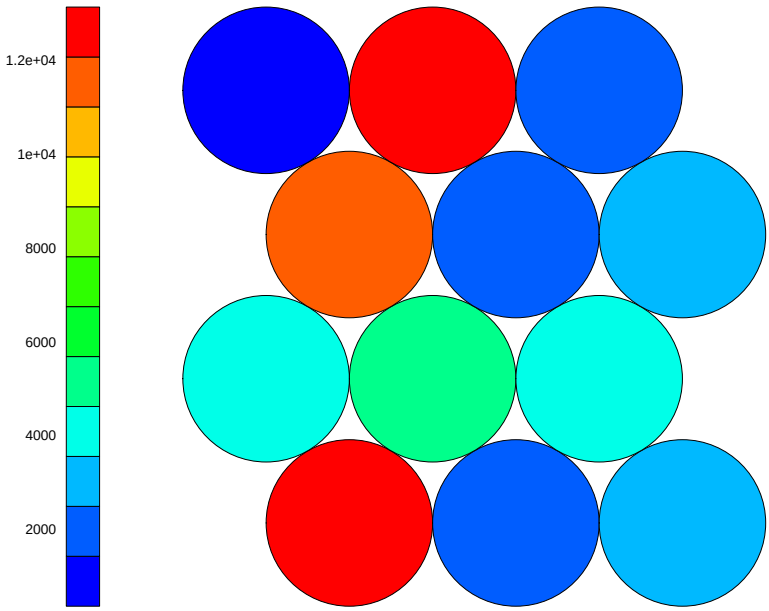
```
plot(som_model,type="dist.neighbours")
```



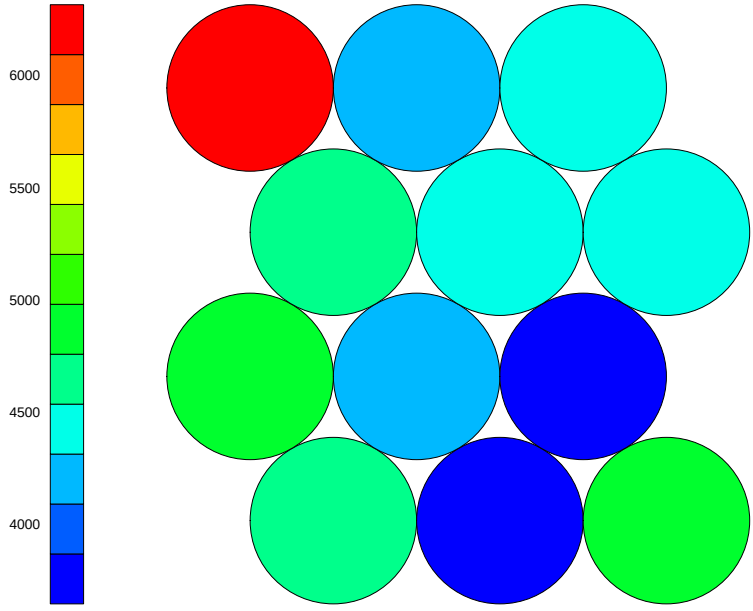
We now plot the componenet plane plots. Which Visualizes the original data over the SOM that we have created.

```
codes <- som_model$codes[[1]]

som_model$data <- data.frame(som_model$data)
for (i in 1:8){
  plot(som_model, type = "property", property = codes[,i], main=names(som_model$data)[i],palette.name=c
  }
```

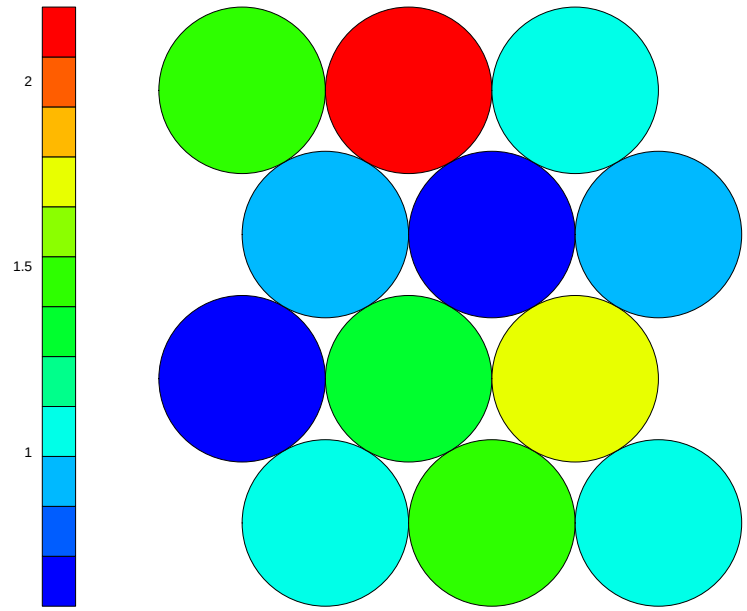
Population



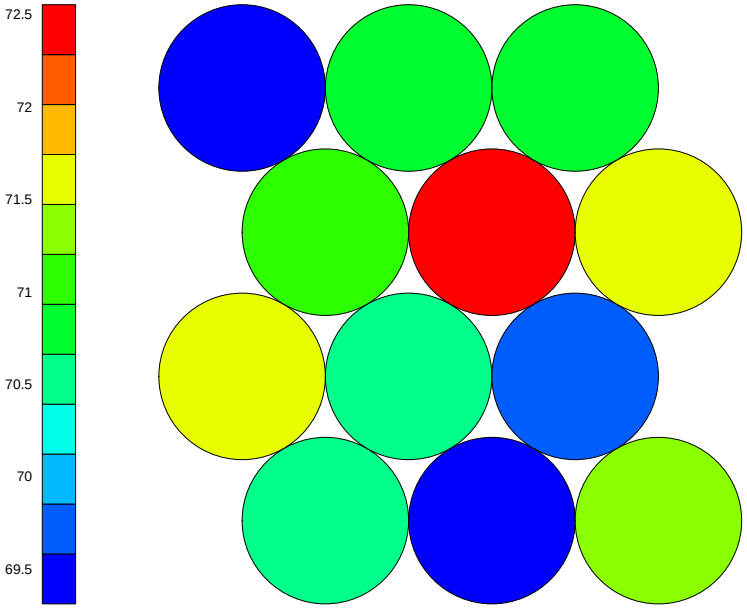
Income



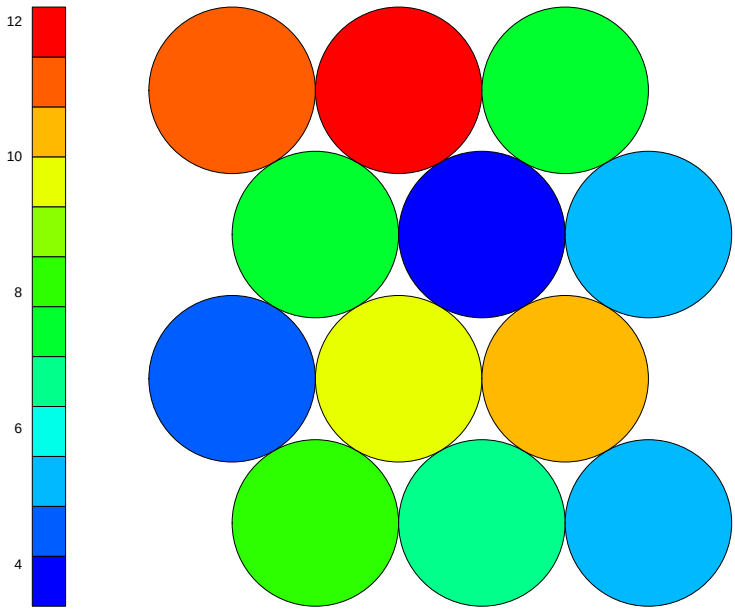
Illiteracy



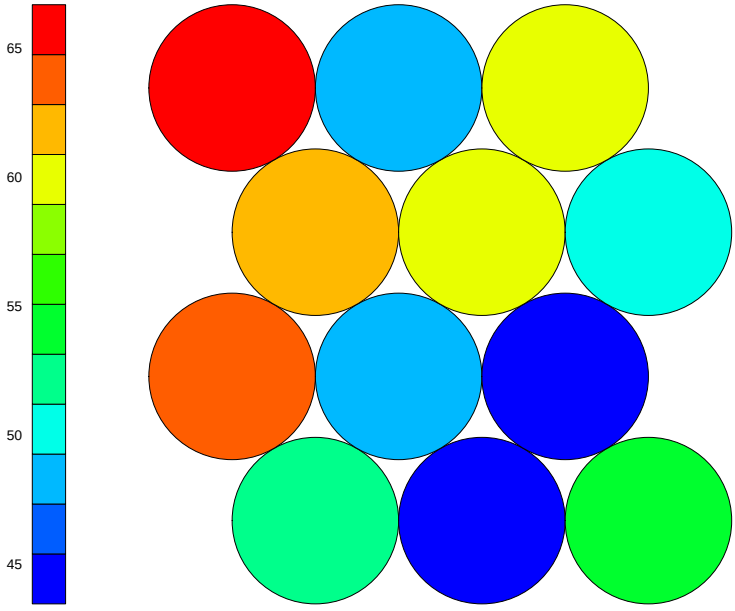
Life.Exp

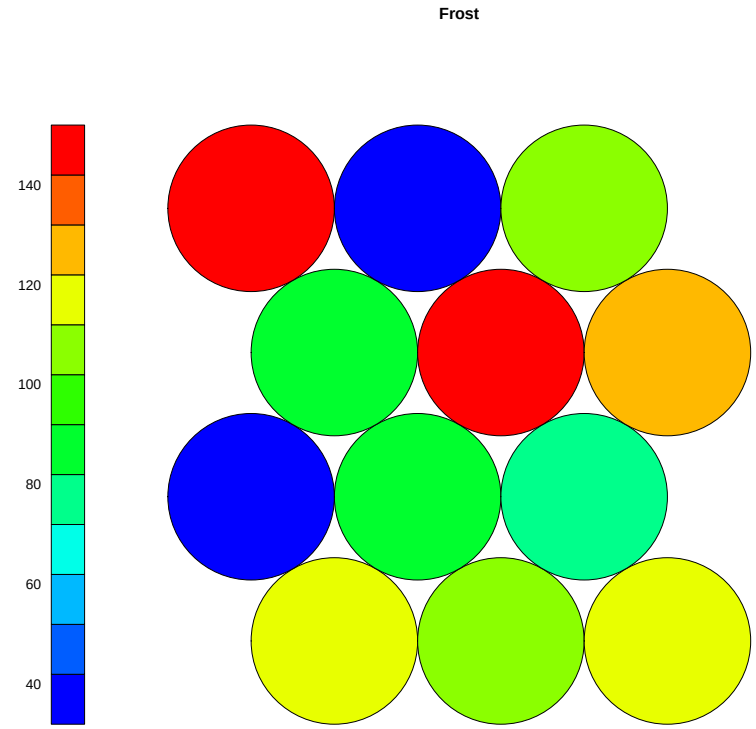


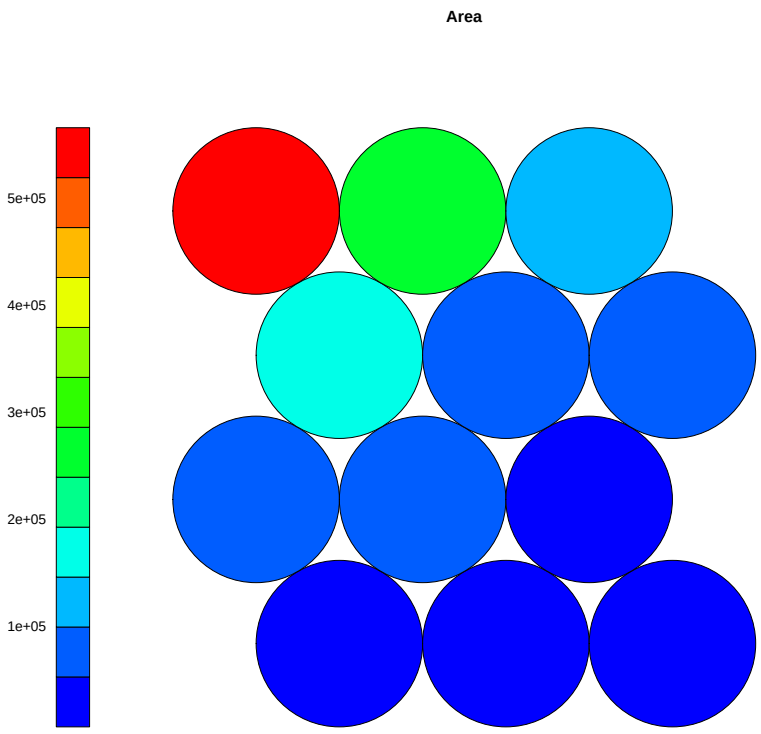
Murder



HS.Grad



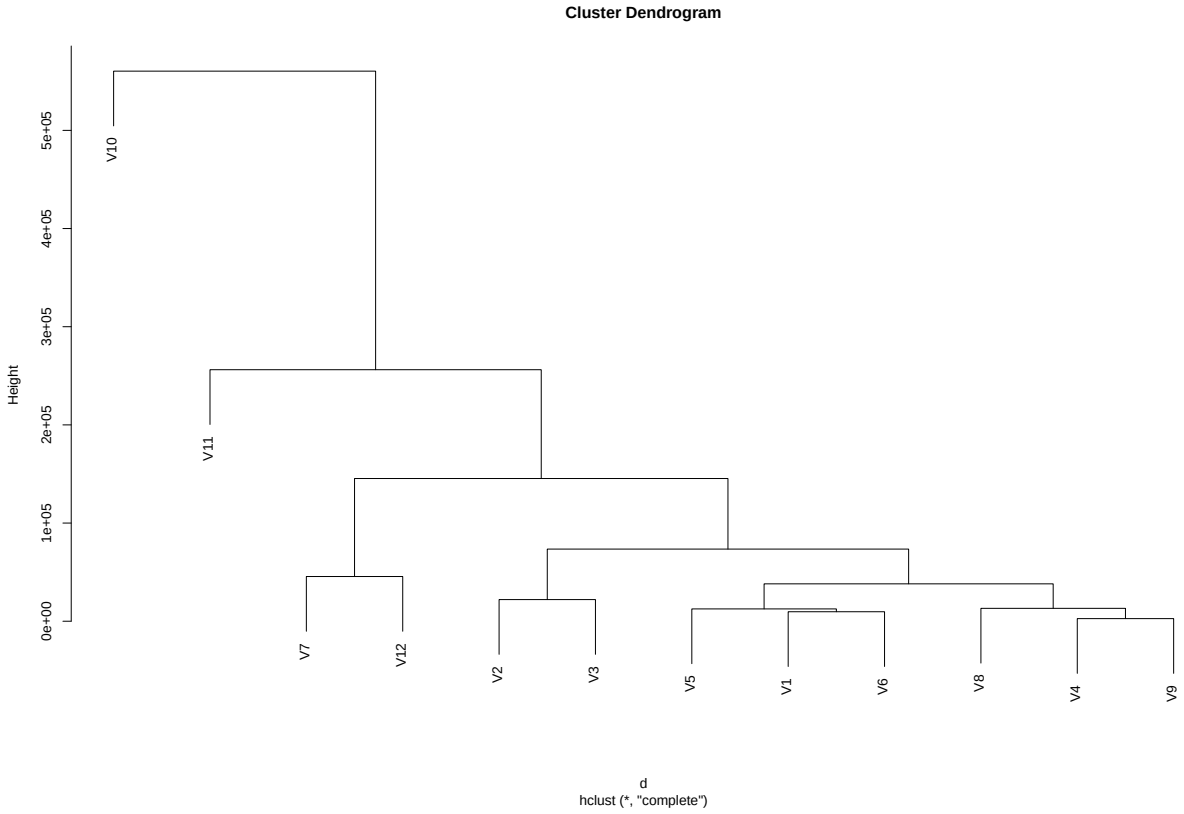




We find the distance using the SOM_Model codes and observe a clear distinction between the height of the

dendrogram between the 2 clusters. This helps us find the point for the cutoff and also helps us easily cluster the dat into 3 distinct groups which was difficult in the case of Plain Heirarchical Clustering using Eucledian Distance.

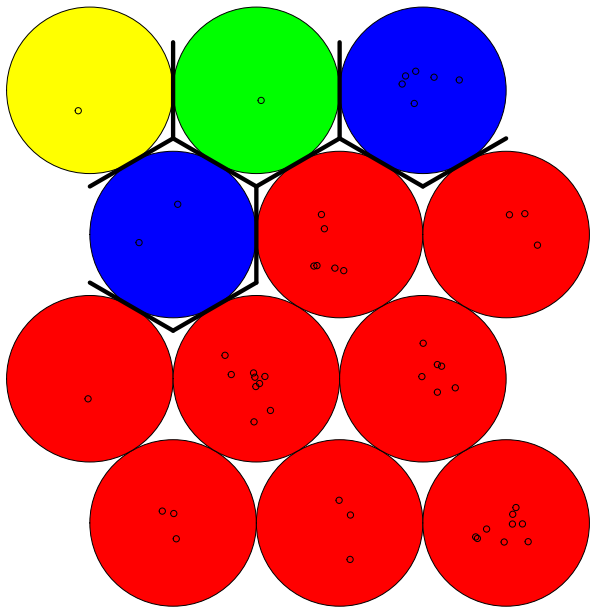
```
d <- dist(codes)
hc <- hclust(d)
plot(hc)
```



We now plot the SOM amd specify the cutoff point at a location where we can observe the approximate cutoff in the Dendrogram plotted above.

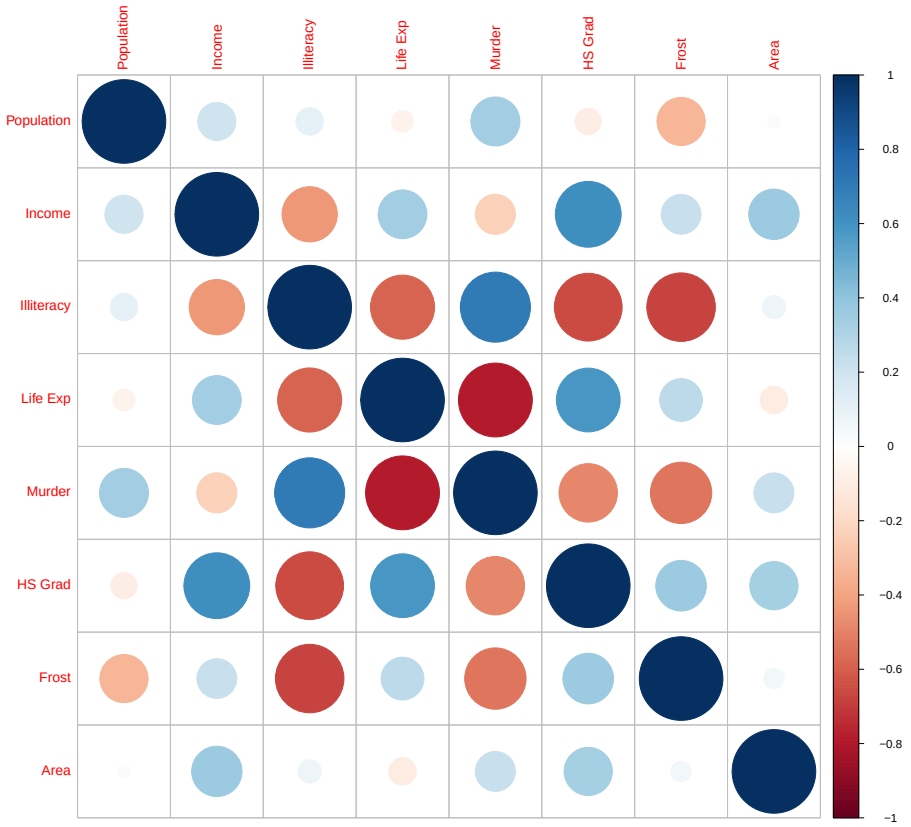
By specifying the cutoff point obtained from the plot above and clustering the data at that height gives us the approximate distribution of the data into 4 clusters.

```
som_cluster <- cutree(hc,k=4)
# plot these results:
my_pal <- c("red","blue","yellow","green")
my_bhcol <- my_pal[som_cluster]

{plot(som_model,type="mapping",col="black",bgcol = my_bhcol)
  add.cluster.boundaries(som_model,som_cluster)}
```



##GAUSSIAN GRAPHICAL MODEL

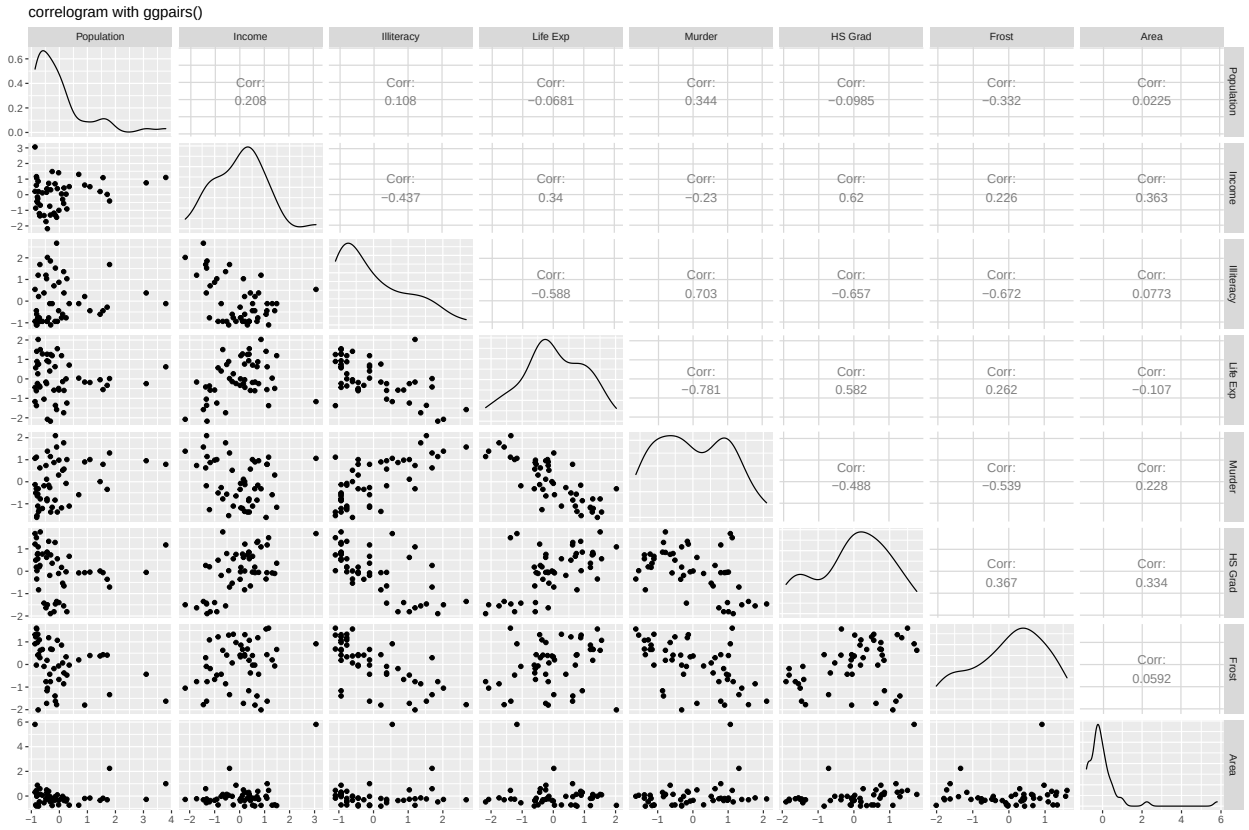
```
M <- cor(df1)
corrplot(M)
```



We observe from the corrplot that Murder and Life Expectency are Negatively correlated. So we remove one of these variables.

We also take a look at the distribution of the data given below.

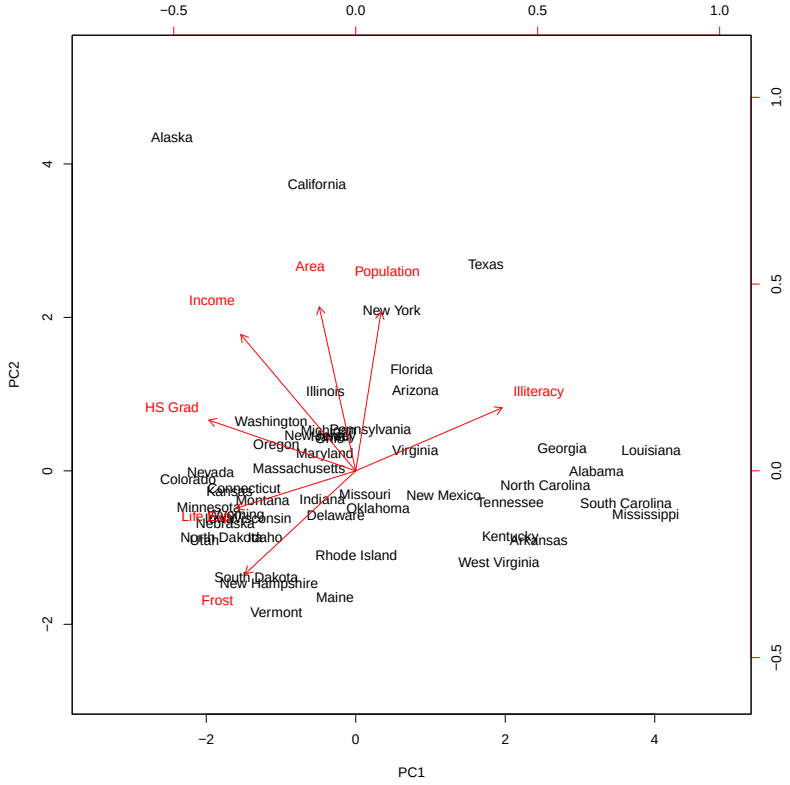
```
df1_1 <- as.data.frame(z)
ggpairs(df1_1, title="correlogram with ggpairs()")
```



We observe that the data is relatively gaussian and we now remove Murder.

```
dats <- z[,-5]
```

```
fit_pca <- prcomp(dats)
xlim_1 <- min(fit_pca$x[,1])-1
xlim_2 <- max(fit_pca$x[,1])+1
ylim_1 <- min(fit_pca$x[,2])-1
ylim_2 <- max(fit_pca$x[,2])+1

biplot(fit_pca,choices = c(1,2),scale = 0,xlim=c(xlim_1,xlim_2),ylim=c(ylim_1,ylim_2))
```



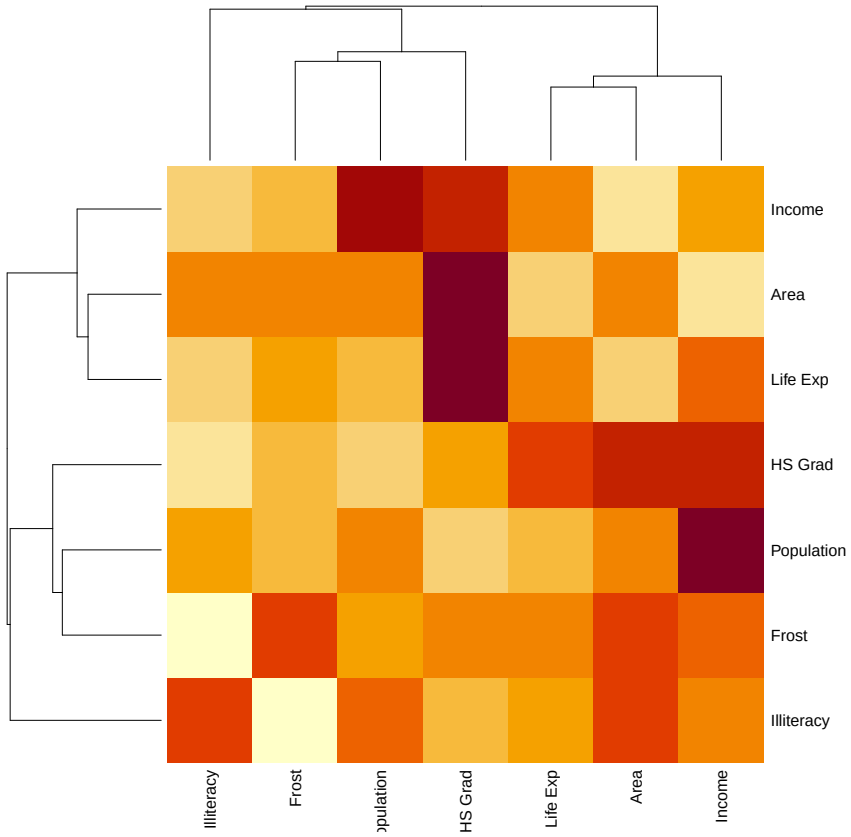
We observe from the Biplot that Alaska, California and Texas are outliers and needs to be removed.

```
new_dats <- dats[-c(2,5,43),]
```

Now we take a look at the partial correlation

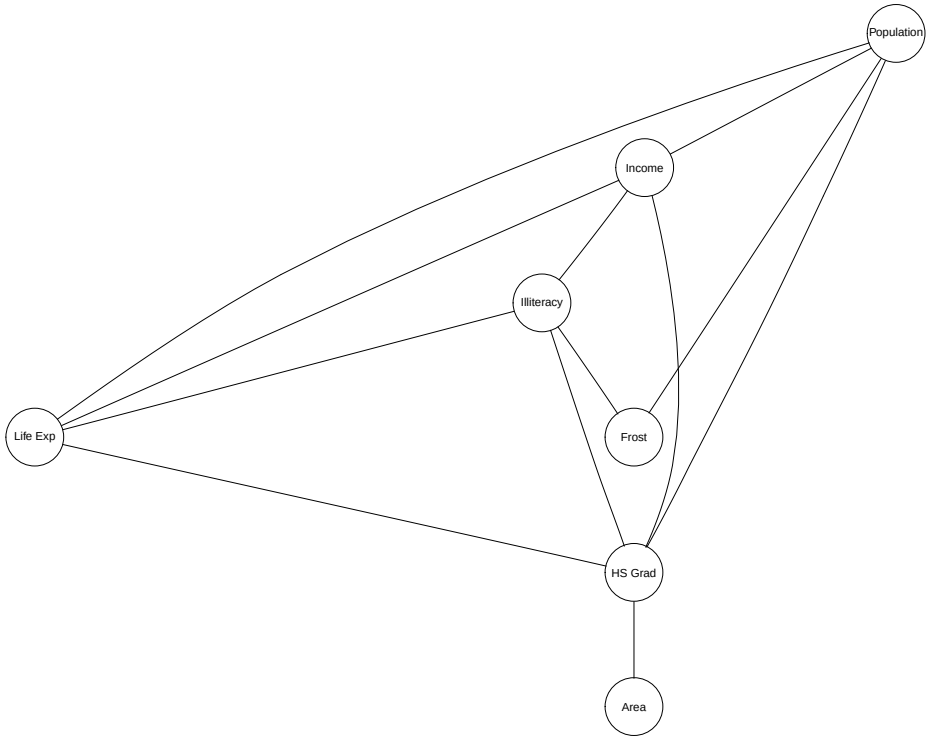
```
part1 <- cov.wt(new_dats, method="ML")
PC.body <- cov2pcor(part1$cov)
diag(PC.body) <- 0
```

```
heatmap(PC.body)
```




```
S <- part1$cov
m0.lasso <- glasso(S,rho=0.1)
my_edges <- m0.lasso$wi!=0
diag(my_edges) <- 0
g.lasso <- as(my_edges,"graphNEL")
nodes(g.lasso) <- colnames(new_dats)
```

```
plot(g.lasso)
```

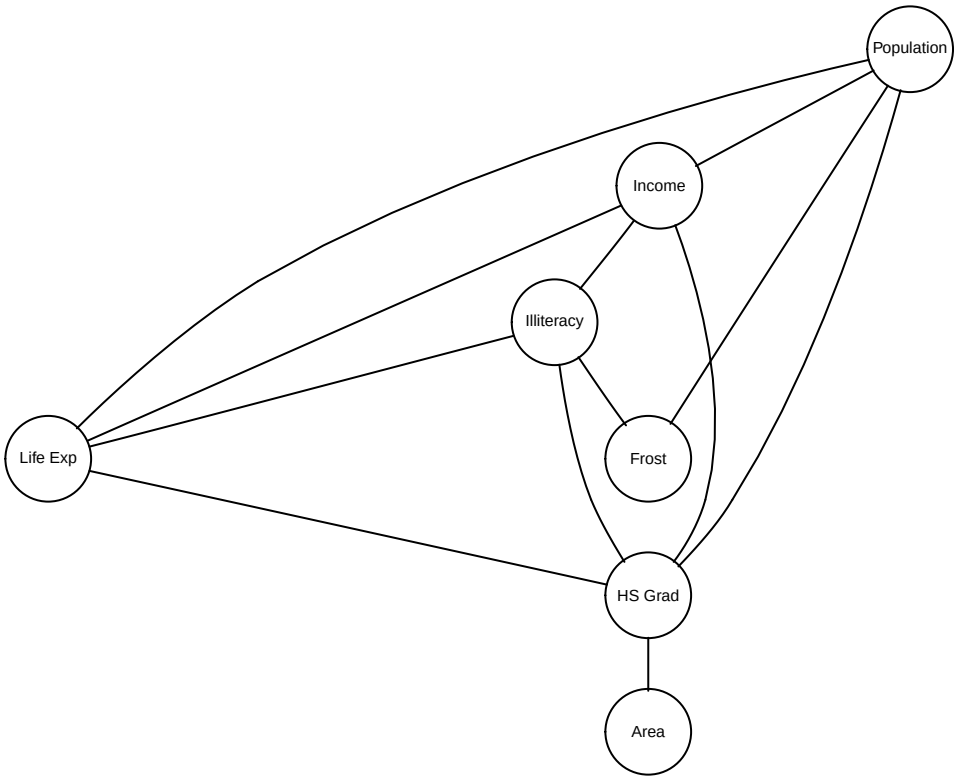


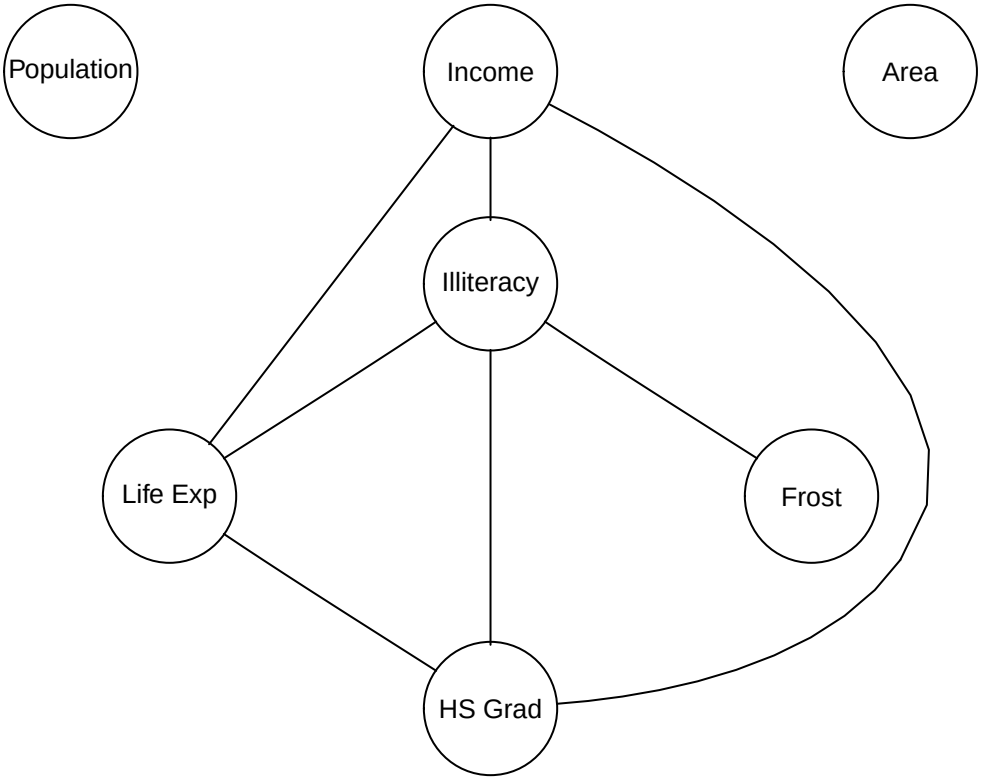

```
my_rhos <- c(0.1,0.2,0.3,0.4,0.5,0.6)
m0.lasso <- glassopath(S,rho=my_rhos)
```

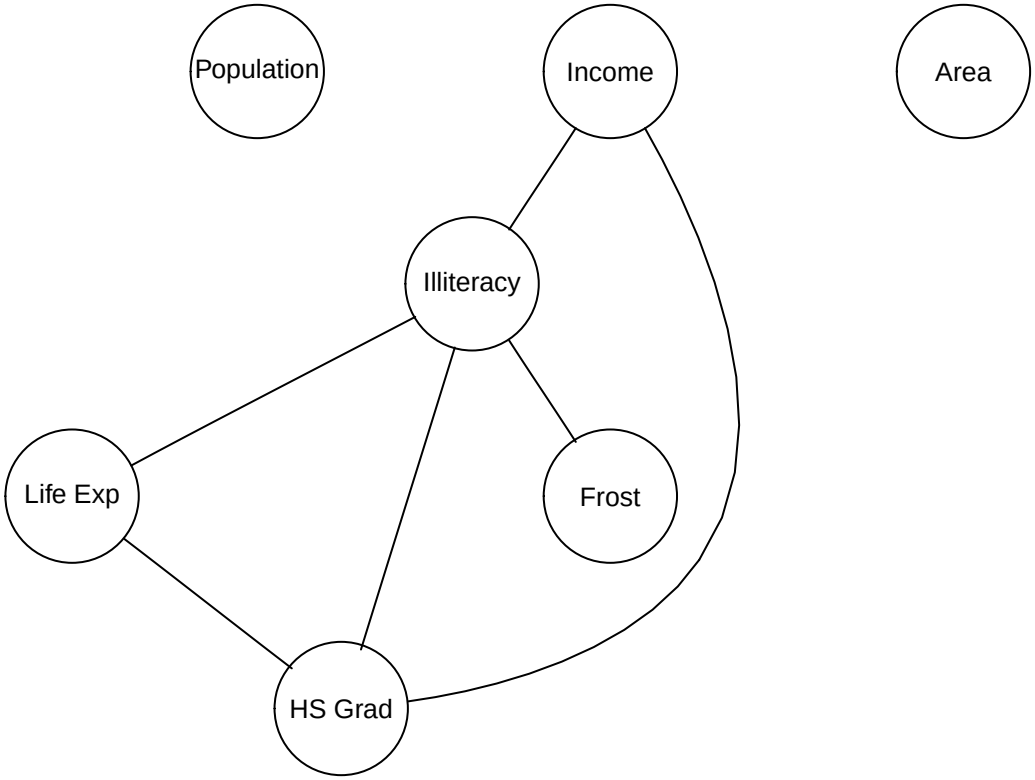
```
## m
## [1] 1
## m
## [1] 2
## m
## [1] 3
## m
## [1] 4
## m
## [1] 1
## m
## [1] 2
## m
## [1] 3
## m
## [1] 4
## rho=
## [1] 0.6
## rho=
## [1] 0.5
## rho=
## [1] 0.4
## rho=
## [1] 0.3
```

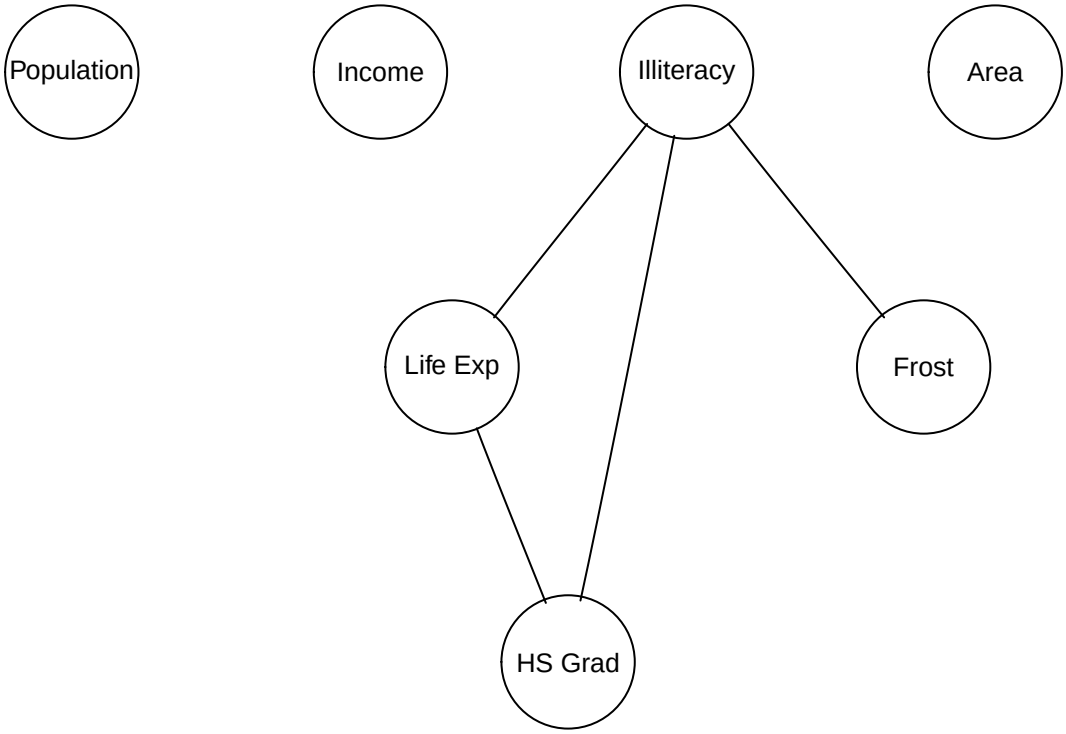
```
## rho=
## [1] 0.2
## rho=
## [1] 0.1
```

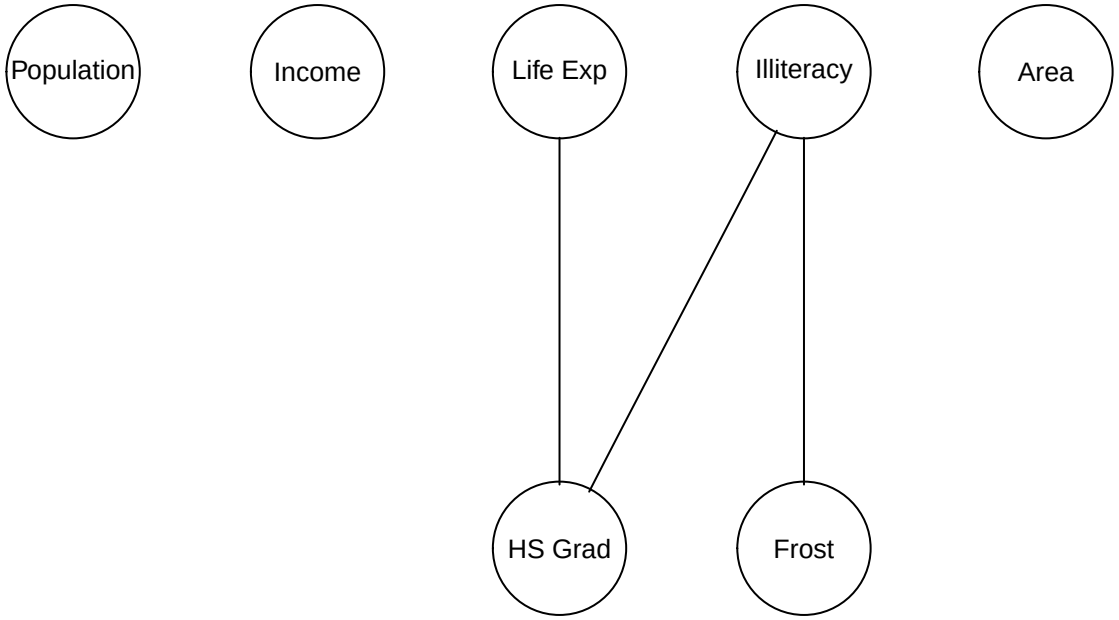
```
for(i in 1:length(my_rhos)){
  my_edges <- m0.lasso$wi[ , , i]!= 0
  diag(my_edges) <- 0
  g.lasso <- as(my_edges,"graphNEL")
  nodes(g.lasso)<-colnames(new_dats)

  plot(g.lasso)
}
```




Population
Income
Area
Illiteracy
Life Exp
Frost
HS Grad


Population
Income
Area
Illiteracy
Life Exp
Frost
HS Grad


Population
Income
Illiteracy
Area
Life Exp
Frost
HS Grad



The Gaussian Graphical Models consider the Principal Components of the data to cluster the data. It uses the covariance and the partial covariance to cluster the data while the Heirarchical Cluster uses the Linkage distance to cluster the factors. The Gaussian Graphical Model can generally map the cause while Hclust is generally used to map the Result. Due to the nature of the mechanism of Gaussian Graphical Models are highly affected by the Skewedness of the data. The data needs to be Normalised before use. Otherwise the Gaussian Graphical Model is unable to cluster the data accurately. Heirarchical cluster is better than the Gaussian Graphical Model in this respect. It is not adversely affected by the nature of distribution of data.