homework4

sai charan talipineni

7 March 2017

Task 1

```
library(cluster)
set.seed(123)
```

##1

```
unempstate_raw<-read.csv("D:/semester/2nd
sem/DATA_MINING/hw4/unempstates.csv")
#summary(unempstate_raw)
#str(unempstate_raw)
head(unempstate_raw)
```

```
##    AL  AK   AZ  AR  CA  CO  CT  DE   FL  GA  HI  ID  IL  IN  IA  KS  KY
## 1 6.4 7.1 10.5 7.3 9.3 5.8 9.4 7.7 10.0 8.3 9.9 5.5 6.4 6.9 4.2 4.3 5.7
## 2 6.3 7.0 10.3 7.2 9.1 5.7 9.3 7.8  9.8 8.2 9.8 5.4 6.4 6.6 4.2 4.2 5.6
## 3 6.1 7.0 10.0 7.1 9.0 5.6 9.2 7.9  9.5 8.1 9.6 5.4 6.4 6.4 4.1 4.2 5.5
## 4 6.0 7.0  9.8 7.0 8.9 5.5 9.1 8.1  9.3 8.0 9.4 5.3 6.4 6.2 4.1 4.2 5.5
## 5 6.0 7.0  9.6 6.9 8.9 5.5 9.0 8.3  9.1 7.9 9.2 5.3 6.5 6.0 4.0 4.2 5.4
## 6 6.0 7.1  9.5 6.8 8.9 5.6 9.0 8.5  9.0 7.9 9.0 5.3 6.6 5.8 4.0 4.1 5.4
##    LA  ME  MD   MA   MI  MN  MS  MO  MT  NE  NV  NH   NJ  NM   NY  NC  ND
## 1 6.2 8.8 6.9 11.1 10.0 6.2 7.0 5.8 5.8 3.6 9.8 7.2 10.5 8.9 10.2 6.7 3.2
## 2 6.2 8.6 6.7 10.9  9.9 6.0 6.8 5.8 5.7 3.5 9.5 7.1 10.4 8.8 10.2 6.5 3.3
## 3 6.2 8.5 6.6 10.6  9.8 5.8 6.6 5.8 5.7 3.3 9.3 7.0 10.4 8.7 10.1 6.3 3.3
## 4 6.3 8.4 6.5 10.3  9.7 5.7 6.4 5.9 5.6 3.2 9.0 6.9 10.3 8.6 10.1 6.2 3.4
## 5 6.4 8.3 6.4 10.1  9.6 5.6 6.3 5.9 5.6 3.0 8.8 6.8 10.3 8.6 10.1 6.0 3.5
## 6 6.6 8.3 6.3  9.8  9.5 5.6 6.3 6.0 5.6 3.0 8.7 6.7 10.3 8.6 10.1 6.0 3.6
##    OH  OK   OR  PA  RI  SC  SD  TN  TX  UT  VT  VA  WA  WV  WI  WY
## 1 8.3 6.4 10.1 8.1 7.8 7.6 3.6 5.9 5.9 6.1 8.8 6.2 8.7 8.3 5.9 4.2
## 2 8.2 6.3  9.8 8.1 7.8 7.4 3.5 5.9 5.9 5.9 8.7 6.1 8.7 8.1 5.7 4.1
## 3 8.0 6.1  9.5 8.0 7.9 7.2 3.4 5.9 5.8 5.7 8.6 5.9 8.7 7.9 5.6 4.0
## 4 7.9 5.9  9.3 8.0 7.9 7.0 3.3 6.0 5.8 5.6 8.6 5.8 8.7 7.6 5.5 3.9
## 5 7.7 5.8  9.1 7.9 7.9 6.9 3.2 6.0 5.8 5.5 8.5 5.7 8.7 7.4 5.4 3.9
## 6 7.6 5.7  9.0 7.9 8.0 6.8 3.2 6.1 5.8 5.4 8.5 5.6 8.8 7.3 5.4 3.9
```

```
t_unemp<-as.data.frame(t(unempstate_raw))
head(t_unemp[,1:50])
```

```
##      V1   V2   V3  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 V17
## AL  6.4  6.3  6.1 6.0 6.0 6.0 6.2 6.3 6.4 6.5 6.6 6.7 6.9 7.0 7.1 7.2 7.2
## AK  7.1  7.0  7.0 7.0 7.0 7.1 7.4 7.7 8.0 8.3 8.5 8.7 8.9 9.1 9.3 9.4 9.6
## AZ 10.5 10.3 10.0 9.8 9.6 9.5 9.5 9.5 9.6 9.6 9.6 9.5 9.4 9.3 9.1 8.9 8.6
## AR  7.3  7.2  7.1 7.0 6.9 6.8 6.7 6.7 6.6 6.6 6.6 6.6 6.5 6.5 6.4 6.3 6.3
```

```
## CA  9.3  9.1  9.0 8.9 8.9 8.9 9.0 9.2 9.3 9.4 9.5 9.4 9.3 9.1 8.9 8.7 8.5
## CO  5.8  5.7  5.6 5.5 5.5 5.6 5.8 6.1 6.3 6.5 6.6 6.6 6.6 6.5 6.4 6.3 6.2
##    V18 V19 V20 V21  V22  V23  V24  V25  V26  V27  V28  V29  V30  V31  V32
## AL 7.1 6.9 6.7 6.5  6.3  6.1  6.0  5.9  5.8  5.7  5.6  5.6  5.7  5.8  6.0
## AK 9.7 9.8 9.8 9.9 10.0 10.2 10.4 10.7 10.8 10.9 10.9 10.8 10.8 10.7 10.6
## AZ 8.4 8.3 8.2 8.1  7.9  7.6  7.3  7.0  6.7  6.4  6.1  5.9  5.7  5.6  5.6
## AR 6.3 6.3 6.4 6.4  6.4  6.4  6.3  6.2  6.1  6.0  5.9  6.0  6.0  6.2  6.3
## CA 8.4 8.3 8.2 8.0  7.9  7.7  7.6  7.5  7.5  7.4  7.4  7.4  7.3  7.2  7.1
## CO 6.2 6.2 6.3 6.4  6.4  6.3  6.2  6.1  5.9  5.8  5.7  5.6  5.5  5.5  5.4
##     V33  V34  V35 V36 V37 V38 V39 V40 V41 V42 V43 V44 V45 V46 V47 V48 V49
## AL  6.1  6.2  6.3 6.3 6.4 6.5 6.6 6.7 6.9 7.0 7.0 7.0 7.0 6.9 6.9 6.9 6.9
## AK 10.5 10.4 10.1 9.9 9.6 9.3 9.1 9.0 8.9 8.9 9.0 9.0 9.1 9.0 8.9 8.8 8.7
## AZ  5.7  5.9  6.0 6.0 5.9 5.7 5.4 5.1 4.9 4.8 4.7 4.8 4.8 4.9 5.0 5.1 5.4
## AR  6.4  6.4  6.4 6.4 6.3 6.3 6.2 6.1 6.0 6.0 6.1 6.1 6.1 6.0 6.0 6.1 6.2
## CA  7.0  6.8  6.7 6.6 6.5 6.4 6.3 6.2 6.2 6.2 6.2 6.2 6.2 6.2 6.2 6.1 6.2
## CO  5.3  5.3  5.1 5.0 4.9 4.7 4.6 4.5 4.5 4.5 4.7 4.8 4.8 4.7 4.5 4.3 4.3
##    V50
## AL 7.1
## AK 8.8
## AZ 5.7
## AR 6.5
## CA 6.3
## CO 4.5
```

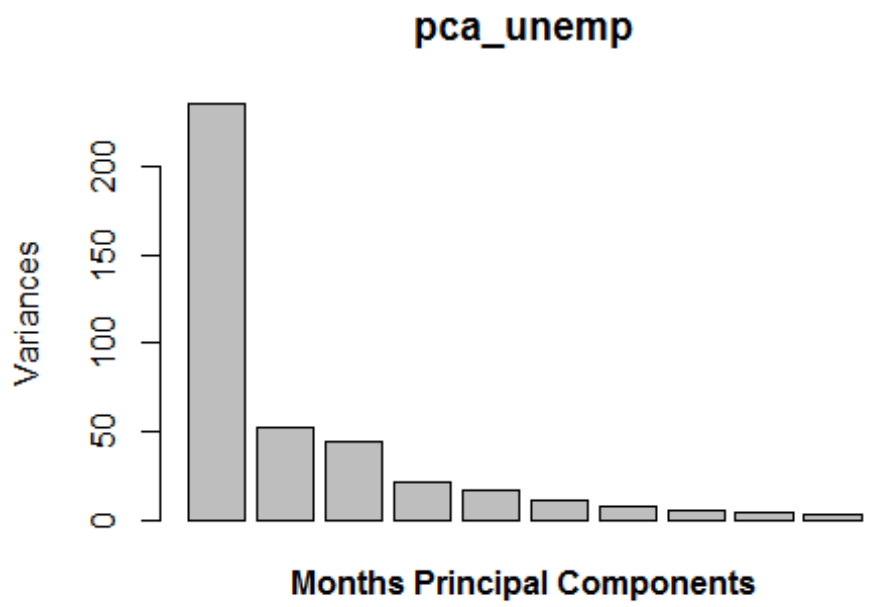
```
pca_unemp<-prcomp(t_unemp,center=TRUE, scale = TRUE)
```

```
summary(pca_unemp)
```

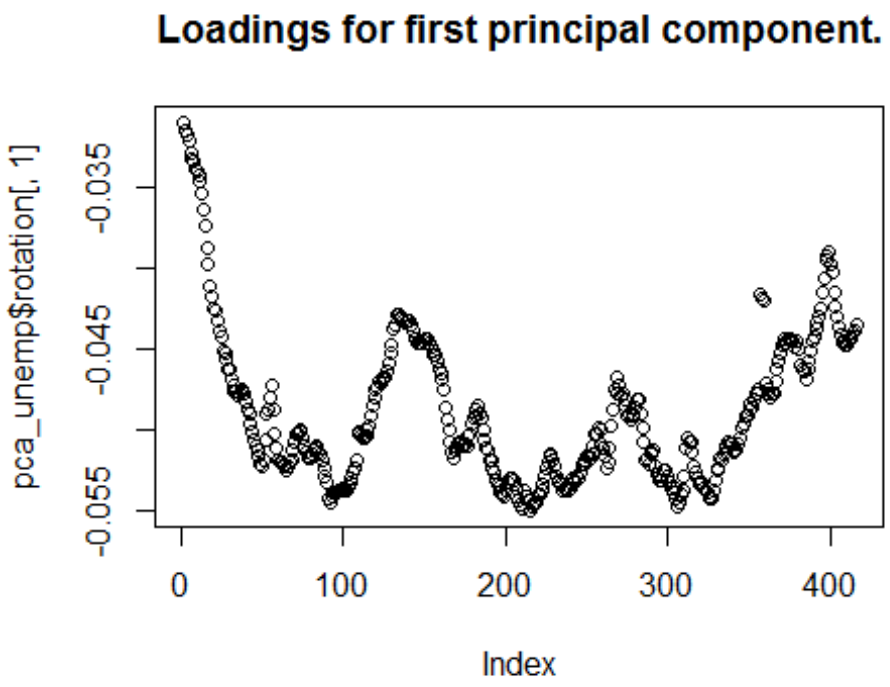
```
## Importance of components:
##                            PC1    PC2    PC3     PC4    PC5     PC6
## Standard deviation     15.3414 7.2506 6.6458 4.56624 4.1048 3.32879
## Proportion of Variance  0.5658 0.1264 0.1062 0.05012 0.0405 0.02664
## Cumulative Proportion   0.5658 0.6921 0.7983 0.84843 0.8889 0.91557
##                            PC7     PC8     PC9    PC10    PC11    PC12
## Standard deviation     2.70779 2.32424 2.00119 1.71616 1.51808 1.21726
## Proportion of Variance 0.01763 0.01299 0.00963 0.00708 0.00554 0.00356
## Cumulative Proportion  0.93320 0.94618 0.95581 0.96289 0.96843 0.97199
##                           PC13    PC14    PC15    PC16    PC17    PC18
## Standard deviation     1.13060 1.07420 1.02820 0.94980 0.92372 0.90282
## Proportion of Variance 0.00307 0.00277 0.00254 0.00217 0.00205 0.00196
## Cumulative Proportion  0.97507 0.97784 0.98038 0.98255 0.98460 0.98656
##                           PC19    PC20    PC21    PC22    PC23    PC24
## Standard deviation     0.82666 0.80240 0.75660 0.66217 0.62099 0.60736
## Proportion of Variance 0.00164 0.00155 0.00138 0.00105 0.00093 0.00089
## Cumulative Proportion  0.98820 0.98975 0.99113 0.99218 0.99311 0.99399
##                           PC25    PC26    PC27    PC28    PC29    PC30
## Standard deviation     0.56712 0.49347 0.46922 0.43814 0.42378 0.40362
## Proportion of Variance 0.00077 0.00059 0.00053 0.00046 0.00043 0.00039
## Cumulative Proportion  0.99477 0.99535 0.99588 0.99634 0.99678 0.99717
##                           PC31    PC32    PC33    PC34    PC35   PC36
```

```
## Standard deviation     0.37850 0.37020 0.33104 0.31422 0.30377 0.2915
## Proportion of Variance 0.00034 0.00033 0.00026 0.00024 0.00022 0.0002
## Cumulative Proportion  0.99751 0.99784 0.99810 0.99834 0.99856 0.9988
##                           PC37    PC38    PC39    PC40    PC41   PC42
## Standard deviation     0.27762 0.26851 0.24522 0.23308 0.21142 0.1994
## Proportion of Variance 0.00019 0.00017 0.00014 0.00013 0.00011 0.0001
## Cumulative Proportion  0.99895 0.99913 0.99927 0.99940 0.99951 0.9996
##                           PC43    PC44    PC45    PC46    PC47    PC48
## Standard deviation     0.18298 0.17705 0.16929 0.15371 0.14039 0.12985
## Proportion of Variance 0.00008 0.00008 0.00007 0.00006 0.00005 0.00004
## Cumulative Proportion  0.99968 0.99976 0.99983 0.99989 0.99993 0.99997
##                           PC49      PC50
## Standard deviation     0.10493 6.813e-15
## Proportion of Variance 0.00003 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00
```

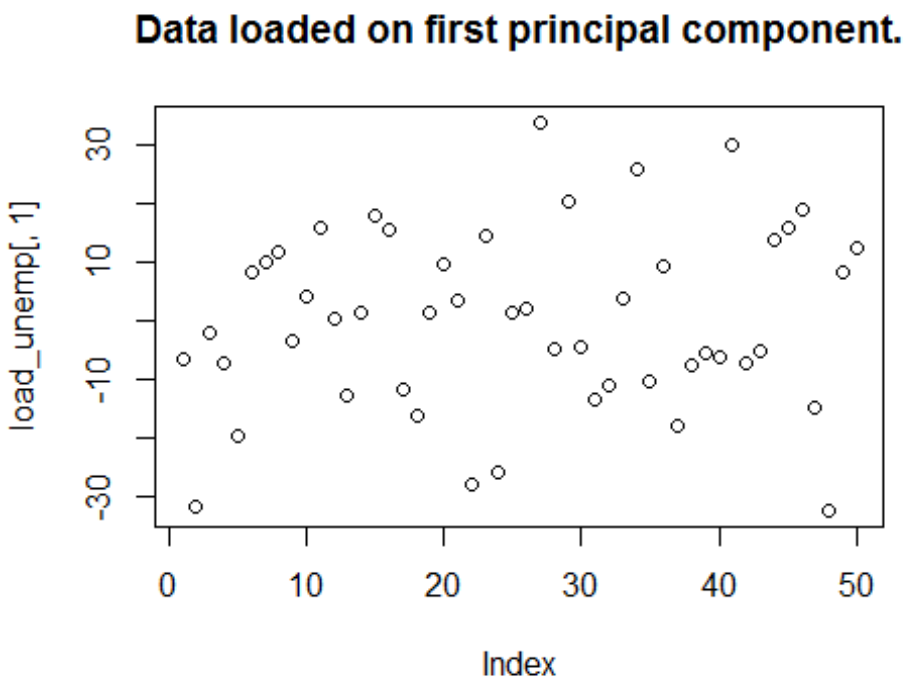
```r
screeplot(pca_unemp)
mtext(side=1, "Months Principal Components",  line=1, font=2)
```



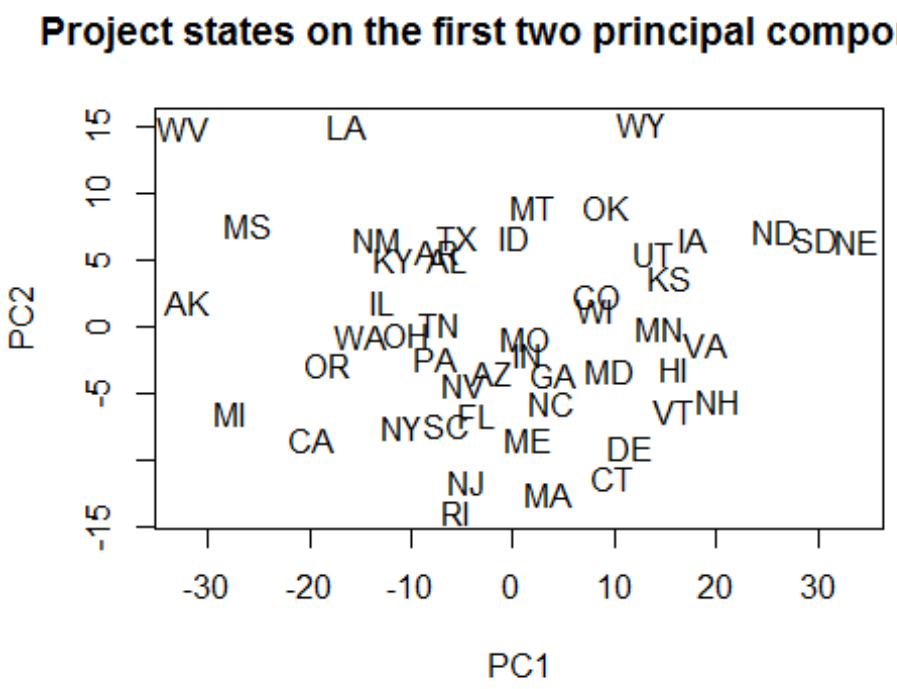

```r
#From the screeplot, principal component 1 contains most of the information.
It is alone sufficient. Based on the requirement of the problem, we can
consider principal components 2 and 3.

plot(pca_unemp$rotation[,1], main = "Loadings for first principal
component.") #pc1
```



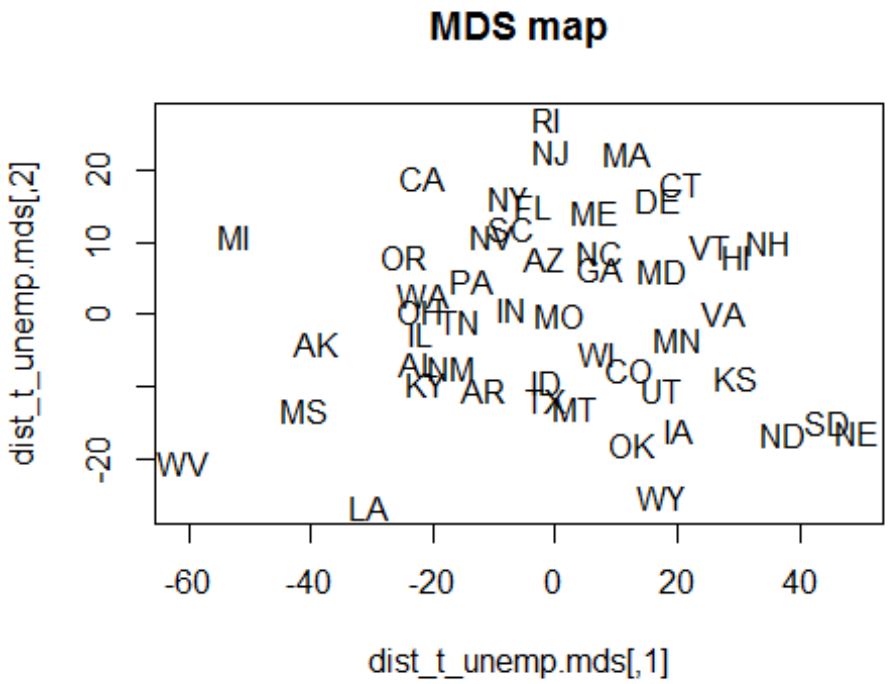

```
load_unemp<-predict(pca_unemp)

plot(load_unemp[,1], main ="Data loaded on first principal component." ) #pc1
on data
```



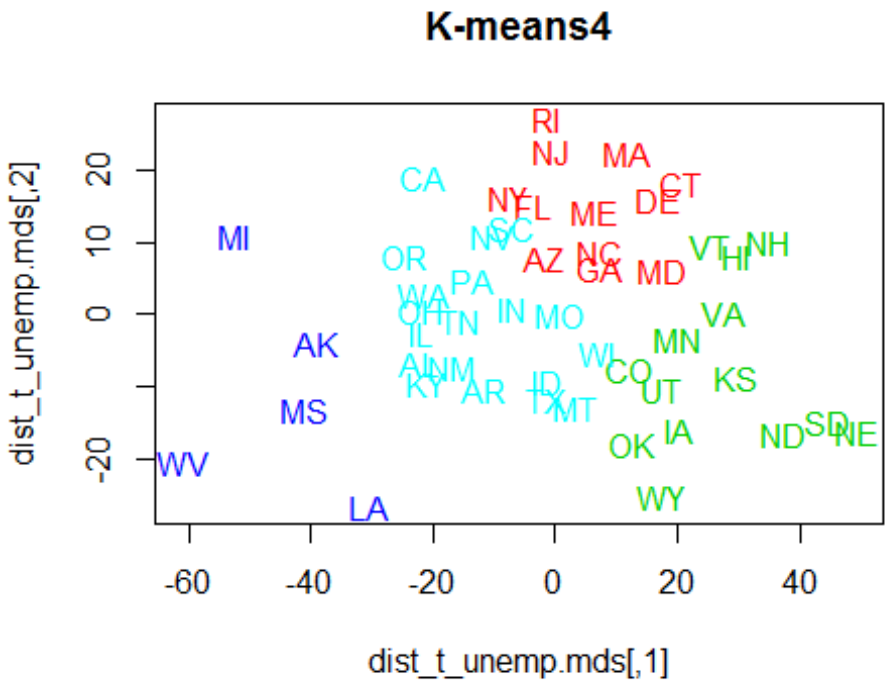

```
##2
plot(load_unemp[,1:2], type="n", main = "Project states on the first two
principal components")
text(x=load_unemp[,1], y=load_unemp[,2], labels=rownames(t_unemp))
```




```r
##3
dist_t_unemp<-dist(t_unemp)
dist_t_unemp.mds<-cmdscale(dist_t_unemp)

plot(dist_t_unemp.mds, type = 'n', main = "MDS map")
text(dist_t_unemp.mds, labels=rownames(t_unemp))
```



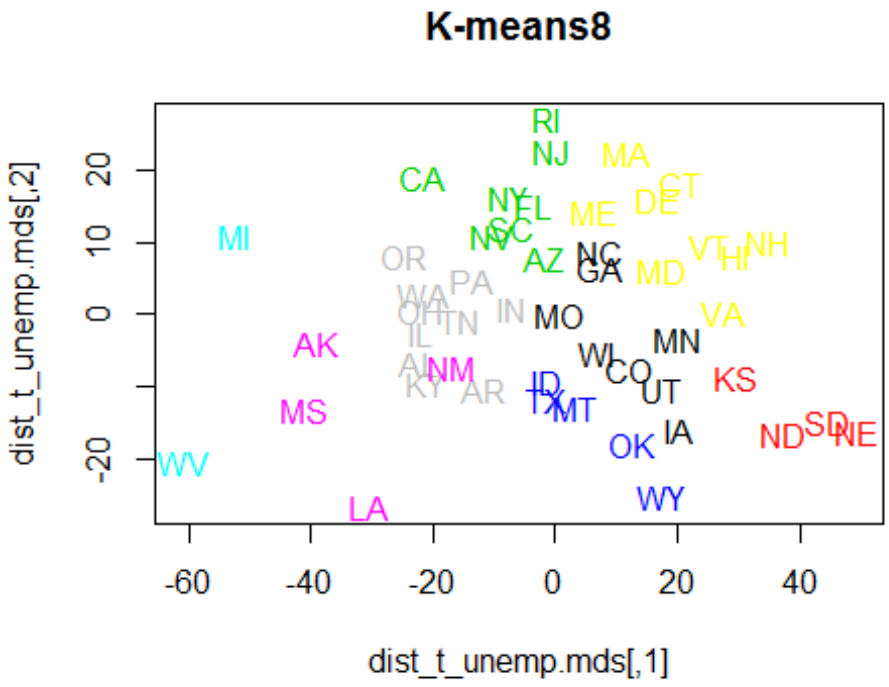

```
##4
state_names<-rownames(t_unemp)

###k-means
kmean4_tunemp <- kmeans(t_unemp, centers=4, nstart=10)
o=order(kmean4_tunemp$cluster)
kmean4_tunemp$cluster[o]

## AZ CT DE FL GA ME MD MA NJ NY NC RI CO HI IA KS MN NE NH ND OK SD UT VT VA
##  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2  2  2  2
## WY AK LA MI MS WV AL AR CA ID IL IN KY MO MT NV NM OH OR PA SC TN TX WA WI
##  2  3  3  3  3  3  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4

#data.frame(state.names[o],kmean4_tunemp$cluster[o])

plot(dist_t_unemp.mds, type = 'n', main = "K-means4")
text(dist_t_unemp.mds, labels=state_names, col = kmean4_tunemp$cluster+1)
```



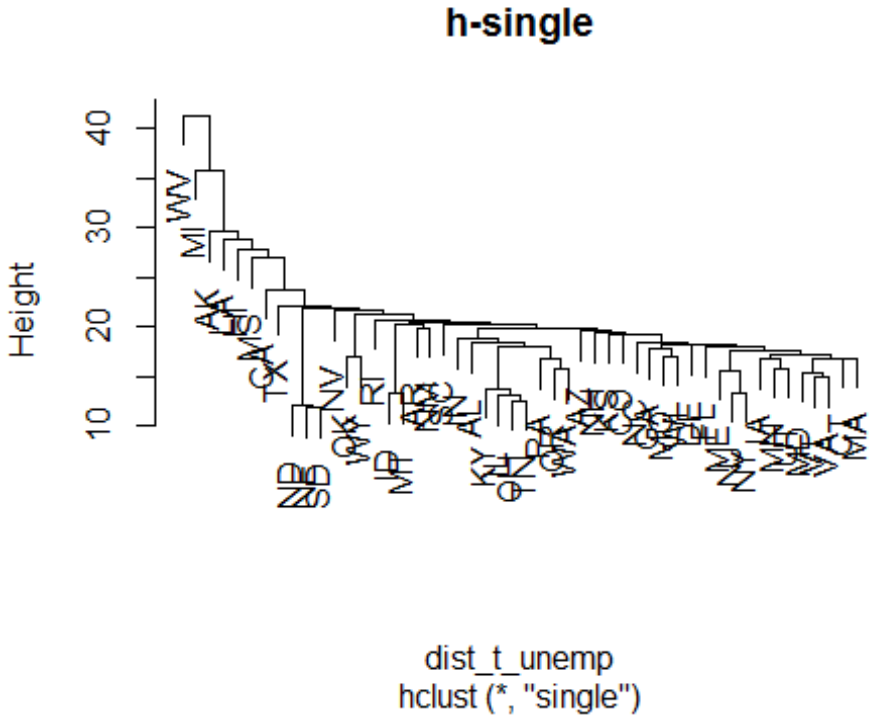

```
kmean8_tunemp <- kmeans(t_unemp, centers=8, nstart=20)
o=order(kmean8_tunemp$cluster)
kmean8_tunemp$cluster[o]
```

```
## KS NE ND SD AZ CA FL NV NJ NY RI SC ID MT OK TX WY MI WV AK LA MS NM CT DE
##  1  1  1  1  2  2  2  2  2  2  2  2  3  3  3  3  3  4  4  5  5  5  5  6  6
## HI ME MD MA NH VT VA AL AR IL IN KY OH OR PA TN WA CO GA IA MN MO NC UT WI
##  6  6  6  6  6  6  6  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8
```

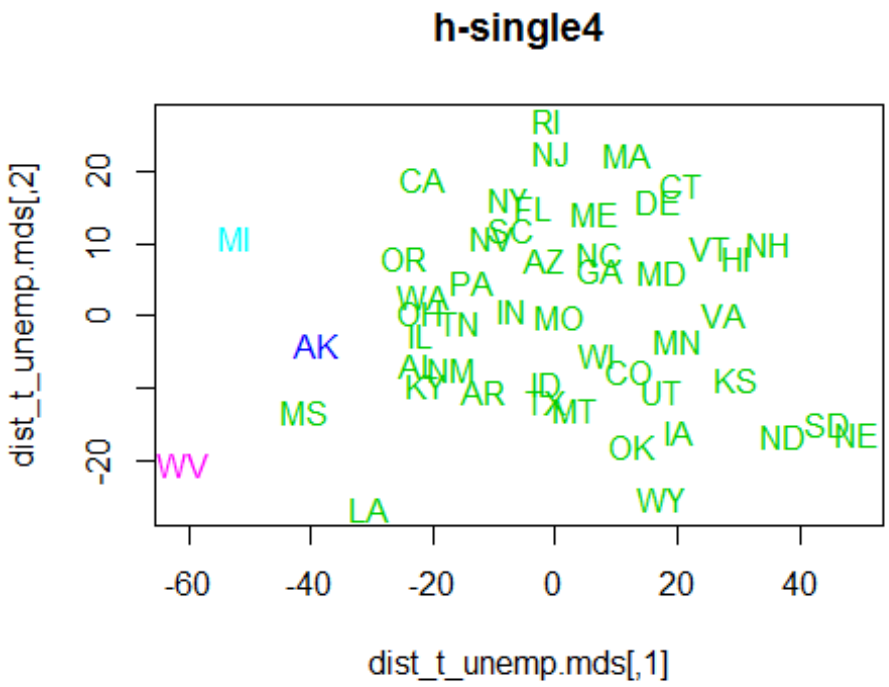
```
plot(dist_t_unemp.mds, type = 'n', main ="K-means8" )
text(dist_t_unemp.mds, labels=state_names, col = kmean8_tunemp$cluster+1)
```



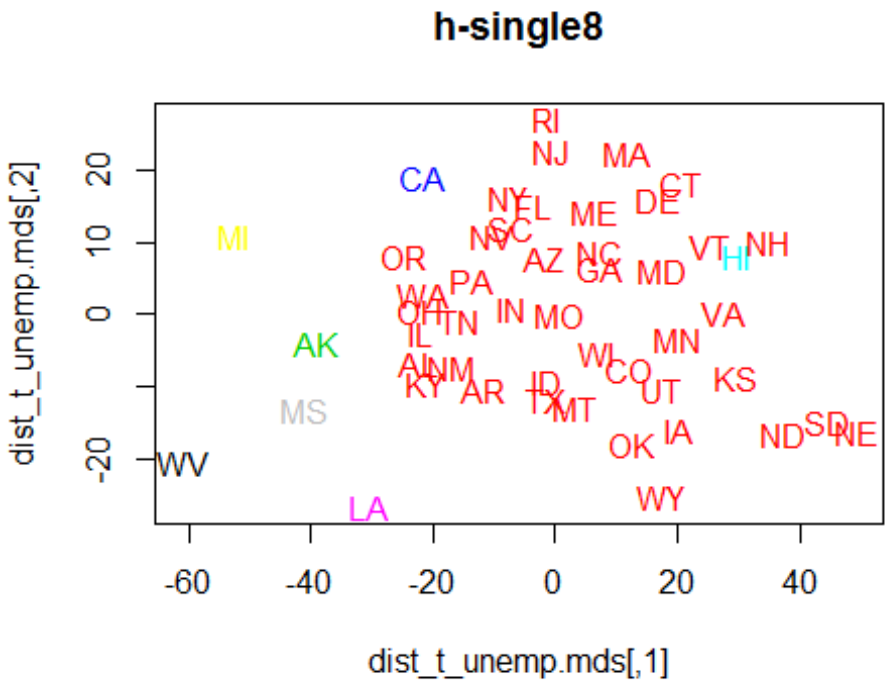

```
###h-clustering
hsingle<-hclust(dist_t_unemp,method = "single")
plot(hsingle, main = "h-single")
```



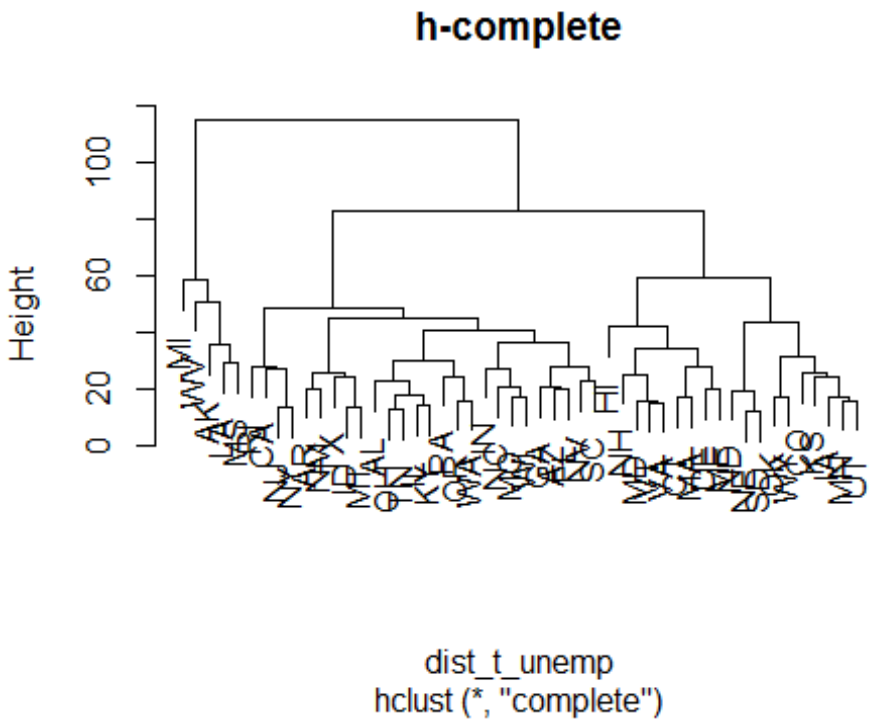
```
hsingle4<-cutree(hsingle,k=4)
plot(dist_t_unemp.mds, type = 'n', main = "h-single4")
text(dist_t_unemp.mds, labels=state_names, col = hsingle4+2)
```



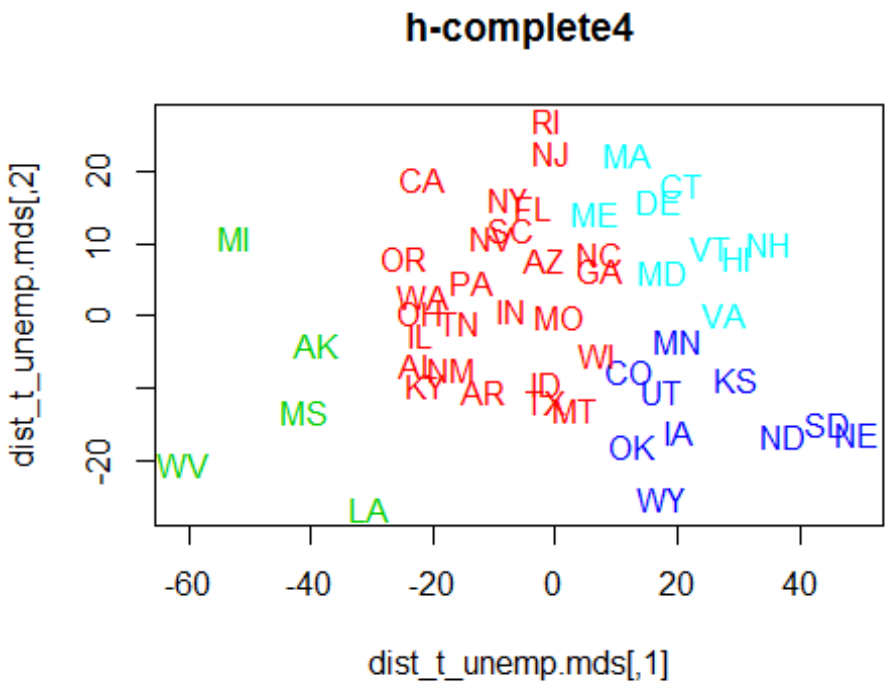

```
hsingle8<-cutree(hsingle,k=8)
plot(dist_t_unemp.mds, type = 'n',main = "h-single8")
text(dist_t_unemp.mds, labels=state_names, col = hsingle8+1)
```



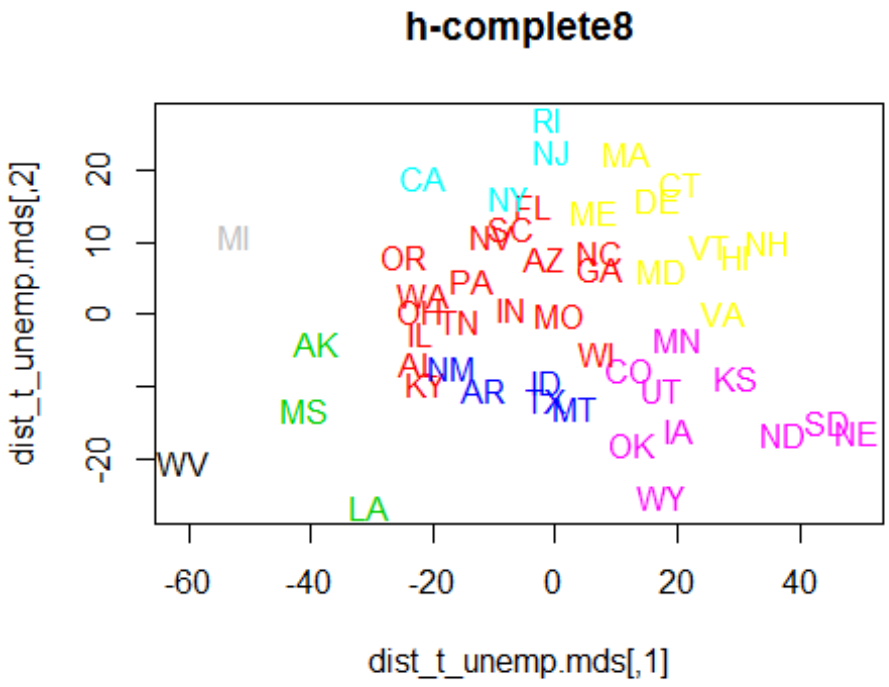

```
hcomplete<-hclust(dist_t_unemp,method = "complete")
plot(hcomplete, main = "h-complete")
```



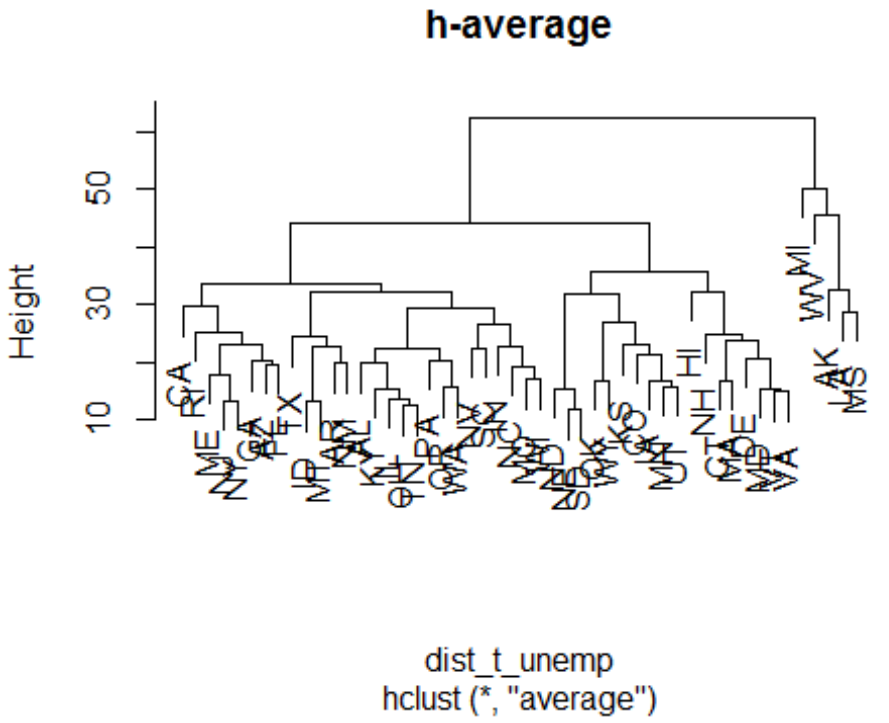
```
hcomplete4<-cutree(hcomplete,k=4)
plot(dist_t_unemp.mds, type = 'n', main = "h-complete4")
text(dist_t_unemp.mds, labels=state_names, col = hcomplete4+1)
```



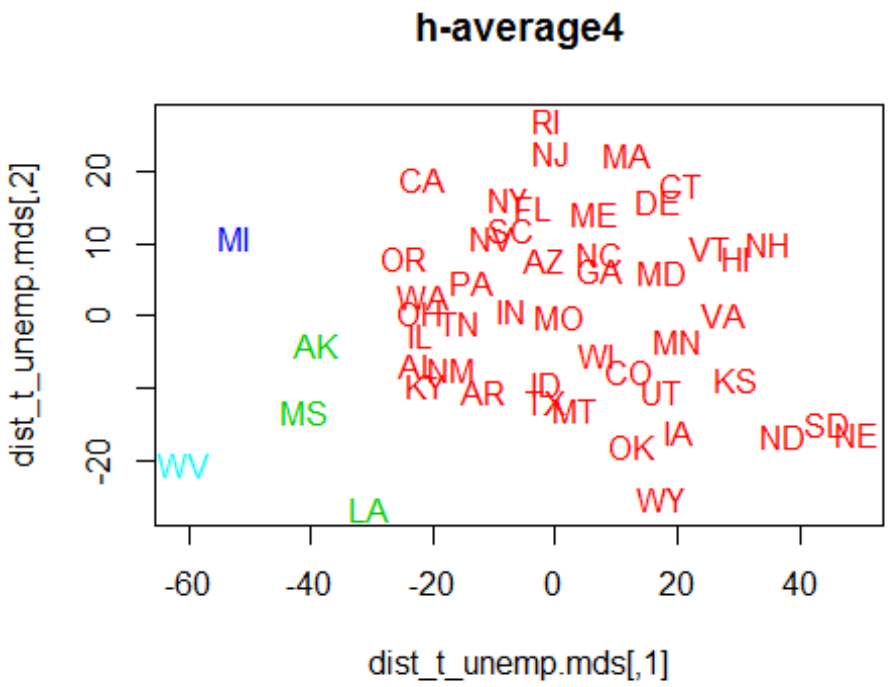

```
hcomplete8<-cutree(hcomplete,k=8)
plot(dist_t_unemp.mds, type = 'n',main = "h-complete8")
text(dist_t_unemp.mds, labels=state_names, col = hcomplete8+1)
```



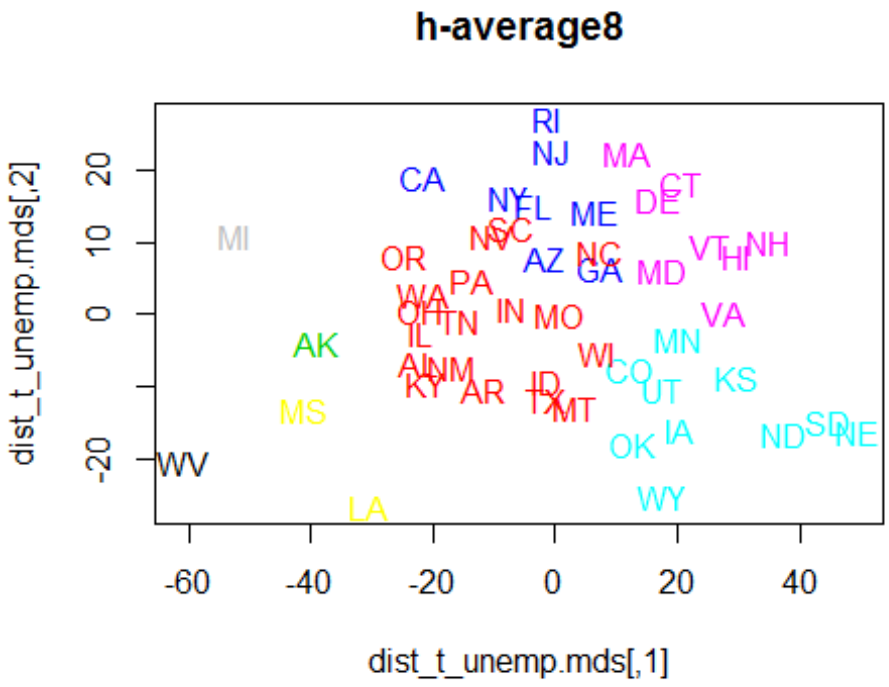

```
haverage<-hclust(dist_t_unemp,method="average")
plot(haverage, main = "h-average")
```



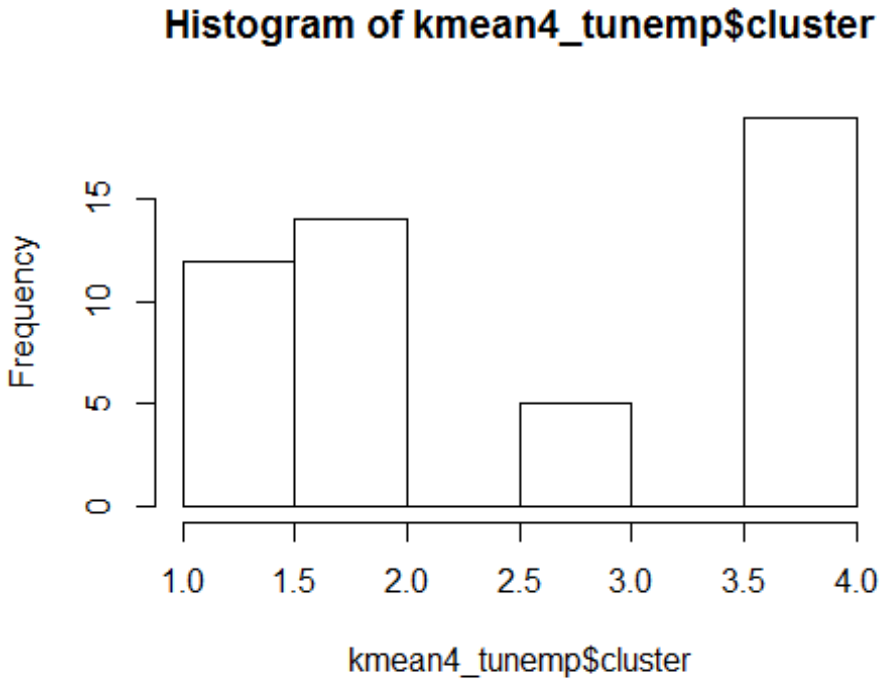
```
haverage4<-cutree(haverage,k=4)
plot(dist_t_unemp.mds, type = 'n', main = "h-average4")
text(dist_t_unemp.mds, labels=state_names, col = haverage4+1)
```



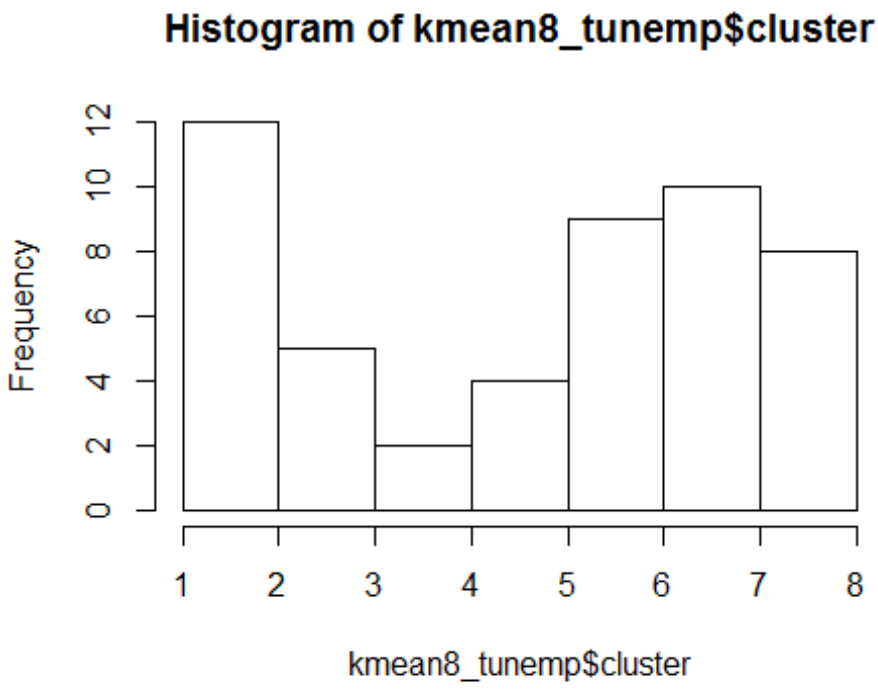

```
haverage8<-cutree(haverage,k=8)
plot(dist_t_unemp.mds, type = 'n', main = "h-average8")
text(dist_t_unemp.mds, labels=state_names, col = haverage8+1)
```



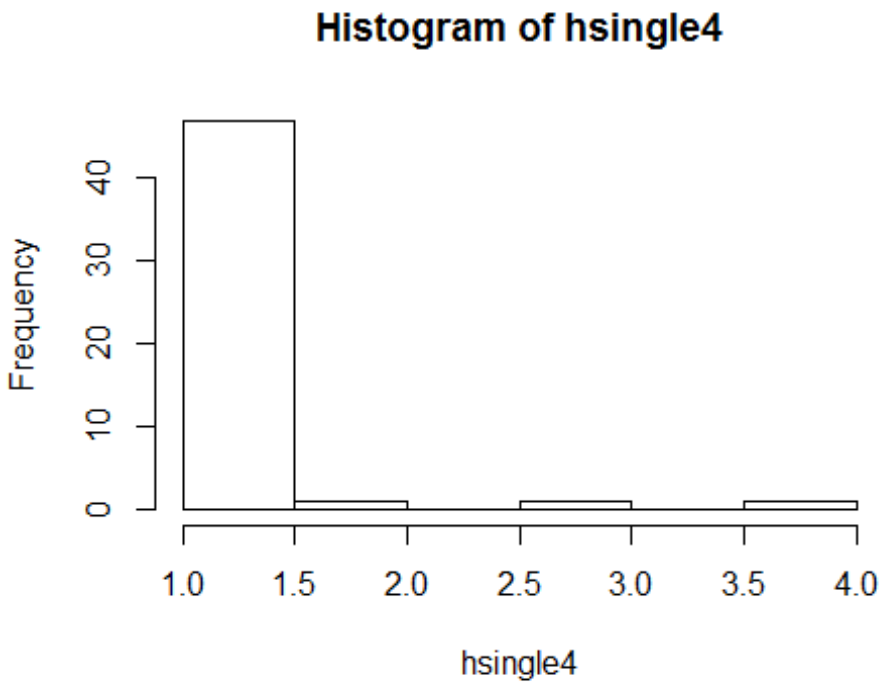

```
##5

hist(kmean4_tunemp$cluster)
```



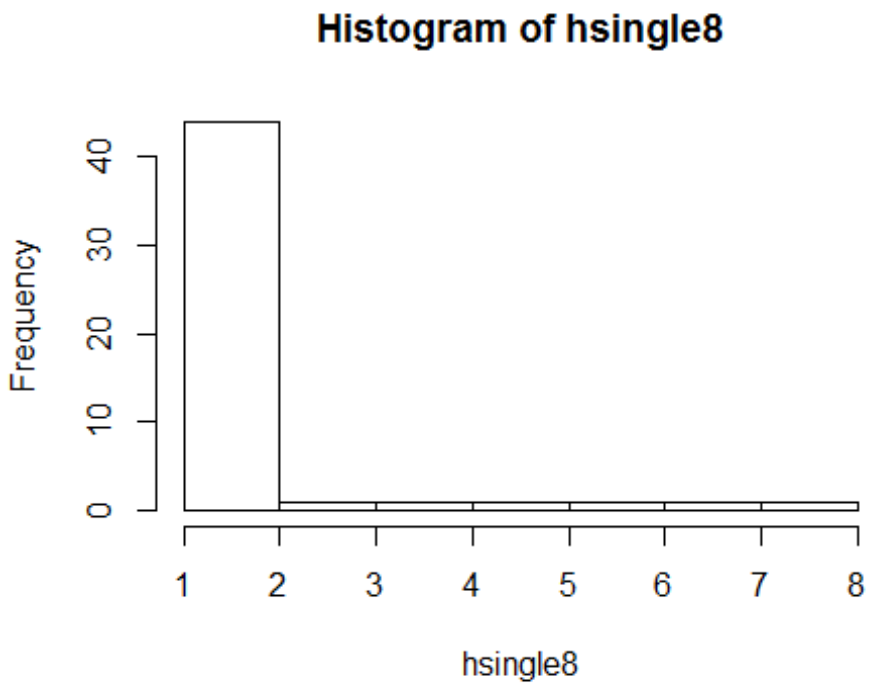
```
hist(kmean8_tunemp$cluster)
```



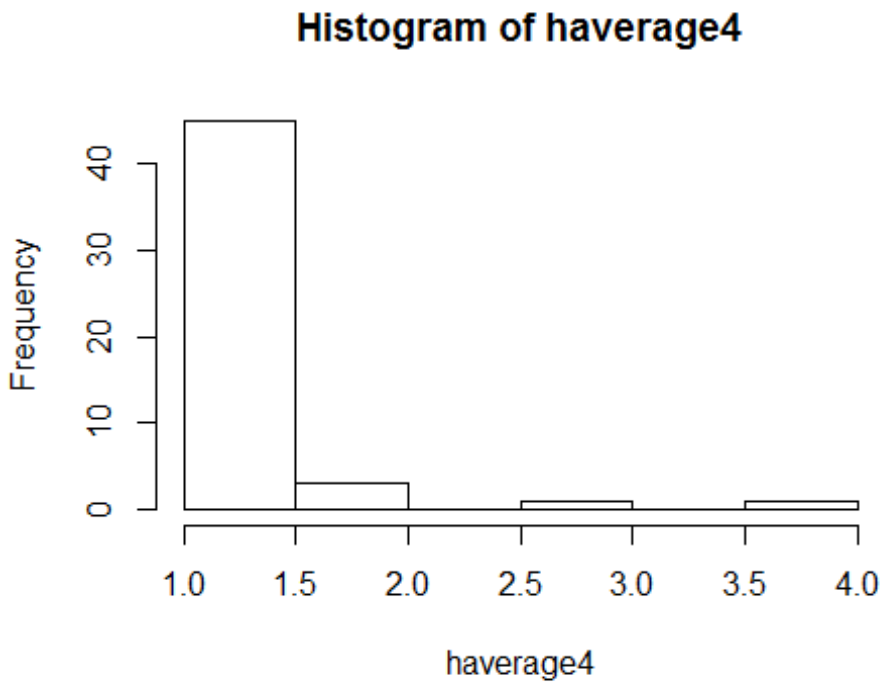

```
hist(hsingle4)
```



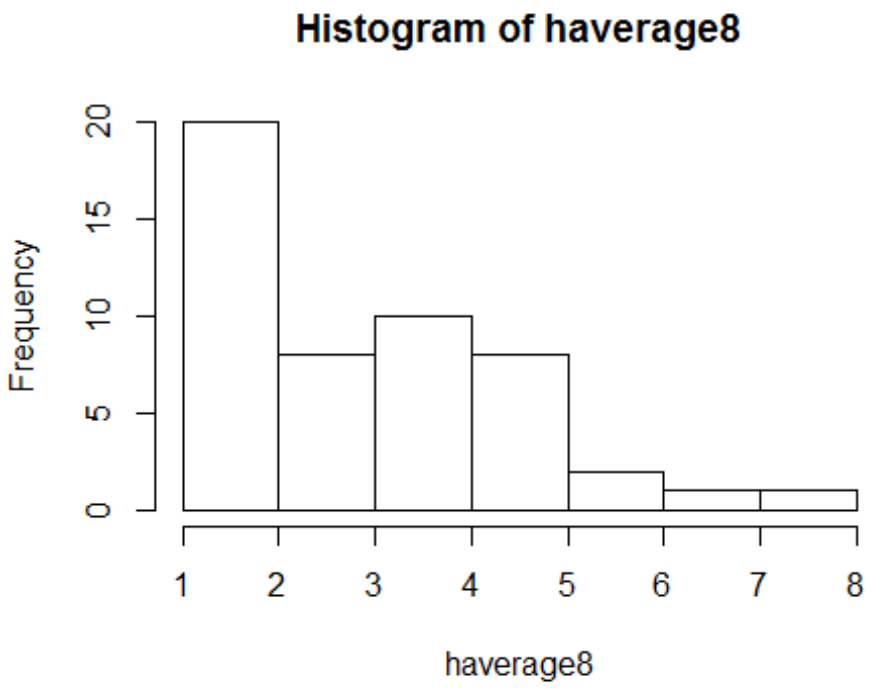
```r
hist(hsingle8)
```



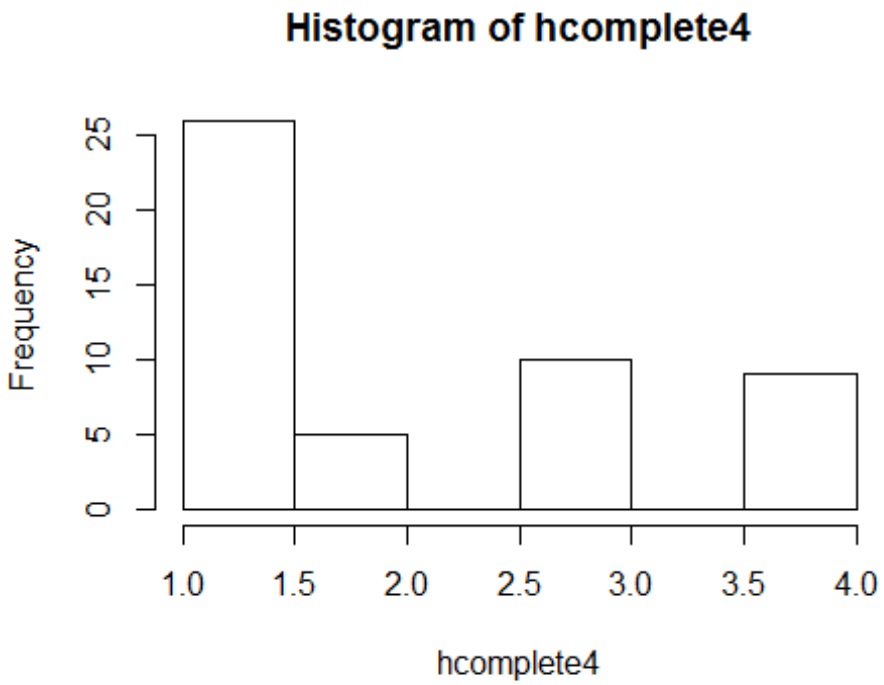

```r
hist(haverage4)
```



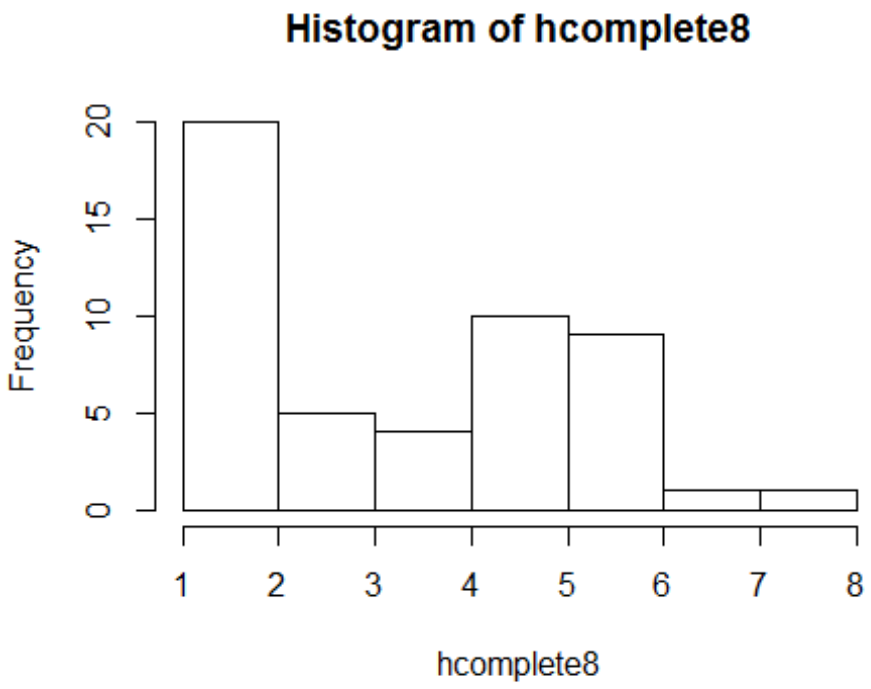
```
hist(haverage8)
```




```
hist(hcomplete4)
```



```
hist(hcomplete8)
```




```
#considering both histograms and  and inter-cluster separation distances the
h-complete 8 clustering result seems to be most meaningful. Based on both
criterias ,the second position could be allocated to the  k-means 4
clustering result.

#Based on histograms only, k-means 4 and k-means 8 are well balanced
clustering results.
```

Task 2

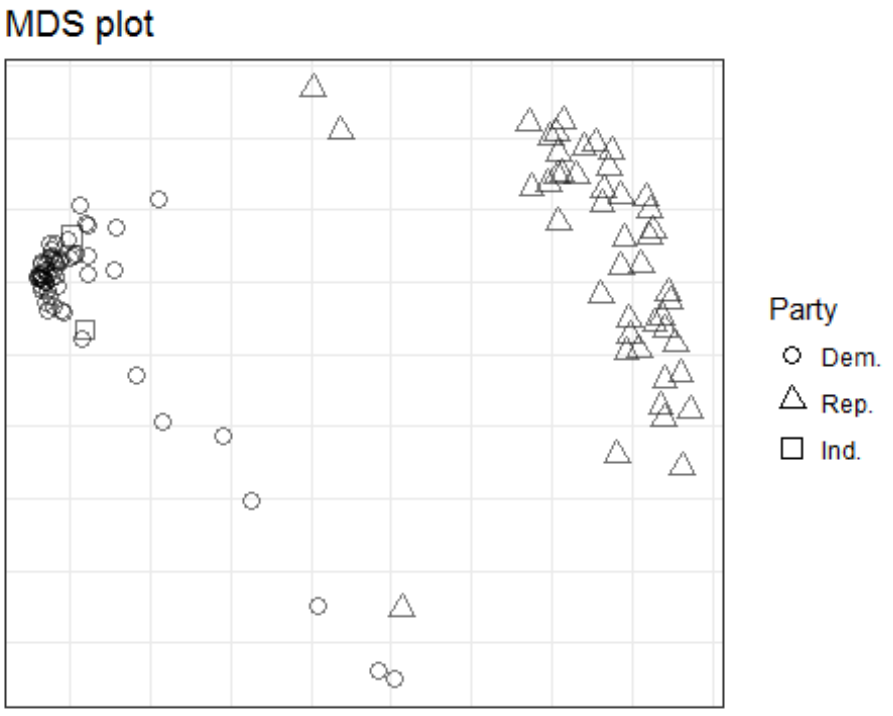
```
library('foreign')
library('ggplot2')

library('dplyr')
```

##1

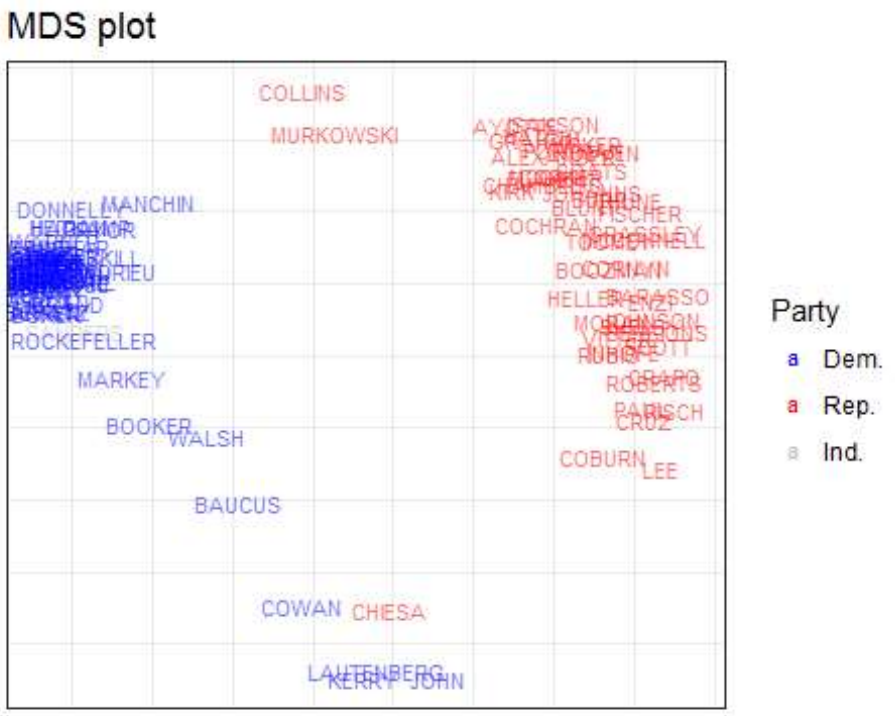
```
## Add all roll call vote data frames to a single list
rollcall.data = read.dta("D:/semester/2nd sem/DATA_MINING/hw4/sen113kh.dta",
convert.factors = FALSE)
dim(rollcall.data)

## [1] 106 666
```

```
rollcall.simplified <- function(df) {
  no.pres <- subset(df, state < 99)
  ## to group all Yea and Nay types together
  for(i in 10:ncol(no.pres)) {
    no.pres[,i] = ifelse(no.pres[,i] > 6, 0, no.pres[,i])
    no.pres[,i] = ifelse(no.pres[,i] > 0 & no.pres[,i] < 4, 1, no.pres[,i])
    no.pres[,i] = ifelse(no.pres[,i] > 1, -1, no.pres[,i])
  }

  return(as.matrix(no.pres[,10:ncol(no.pres)]))
}

rollcall.simple = rollcall.simplified(rollcall.data)

## Multiply the matrix by its transpose to get Senator-to-Senator
tranformation,
## and calculate the Euclidan distance between each Senator.
rollcall.dist = dist(rollcall.simple %*% t(rollcall.simple))

## Do the MDS
rollcall.mds = as.data.frame((cmdscale(rollcall.dist, k = 2)) * -1)


congresses = 113

  names(rollcall.mds) = c("x", "y")

  congress = subset(rollcall.data, state < 99)

  congress.names = congress$name

  rollcall.mds = transform(rollcall.mds, name = congress.names, party =
as.factor(congress$party), congress = congresses)


head(rollcall.mds)

##            x          y        name party congress
## 2  4396.1739 -165.20565 SESSIONS      200      113
## 3  4265.6878 -141.72800 SHELBY        200      113
## 4   369.3607  523.14959 MURKOWSKI     200      113
## 5 -2777.7791   29.33102 BEGICH        100      113
## 6  3061.9752  368.41034 FLAKE         200      113
## 7  3113.7826  376.58455 MCCAIN        200      113

cong.113 <- rollcall.mds

base.113 <- ggplot(cong.113, aes(x = x, y = y)) +
  scale_alpha(guide="none") + theme_bw() +
```

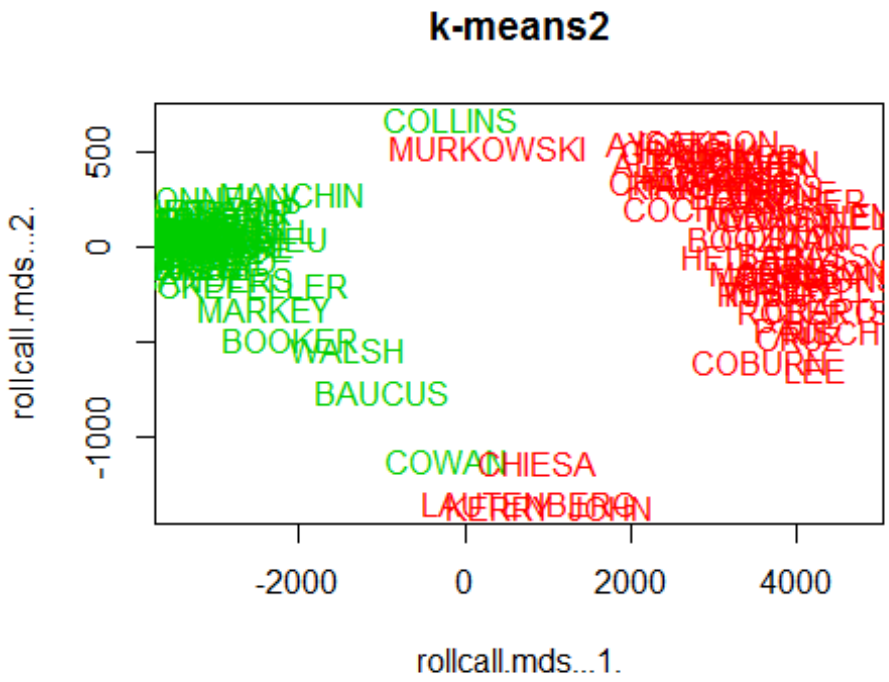
```
  theme(axis.ticks = element_blank(),
        axis.text.x = element_blank(),
        axis.text.y = element_blank()) +
  xlab("") +
  ylab("") +
  ggtitle("MDS plot")+
  scale_shape(name = "Party", breaks = c("100", "200", "328"),
              labels = c("Dem.", "Rep.", "Ind."), solid = FALSE) +
  scale_color_manual(name = "Party", values = c("100" = "blue",
                                                "200" = "red",
                                                "328"="grey"),
                     breaks = c("100", "200", "328"),
                     labels = c("Dem.", "Rep.", "Ind."))

print(base.113 + geom_point(aes(shape = party,
                                alpha = 0.75),size=3))
```



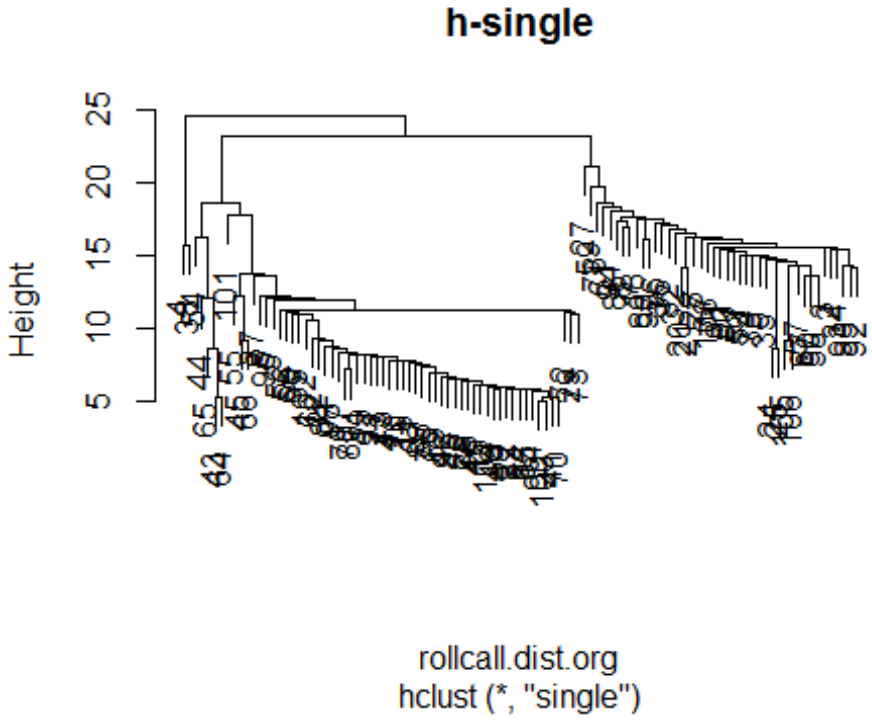

```
print(base.113 + geom_text(aes(color = party,  alpha = 1, label =
cong.113$name),size=3))
```



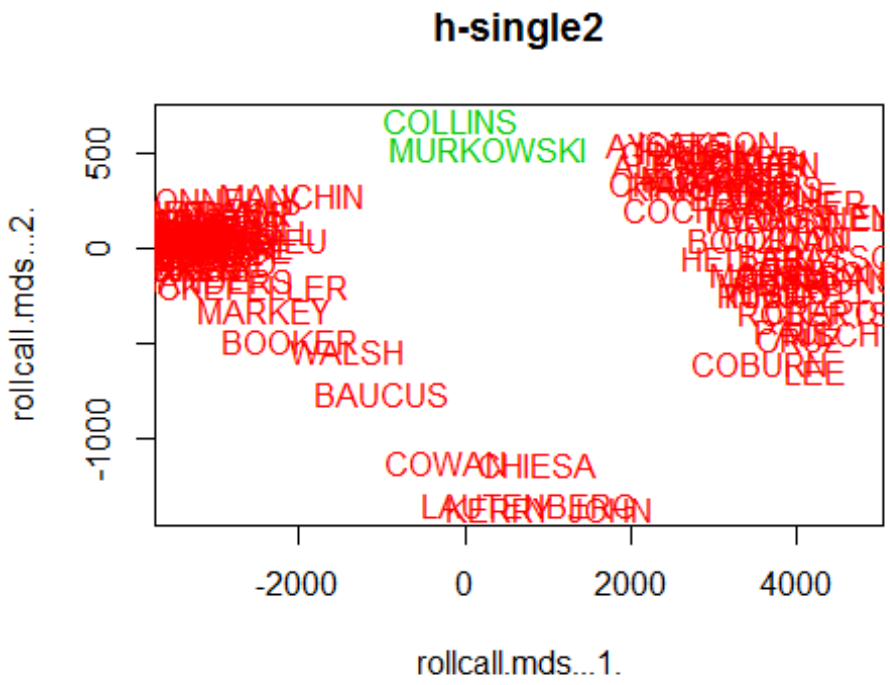

```
##2
rollcall.mds2<-data.frame(rollcall.mds[,1],rollcall.mds[,2])
rollcall.dist.org<-dist(rollcall.simple)

#k-means
kmeans2_rollcall<-kmeans(rollcall.simple,centers = 2,nstart = 10)
plot(rollcall.mds2, type = 'n',main="k-means2")
text(rollcall.mds2, labels=rollcall.mds$name, col =
kmeans2_rollcall$cluster+1)
```



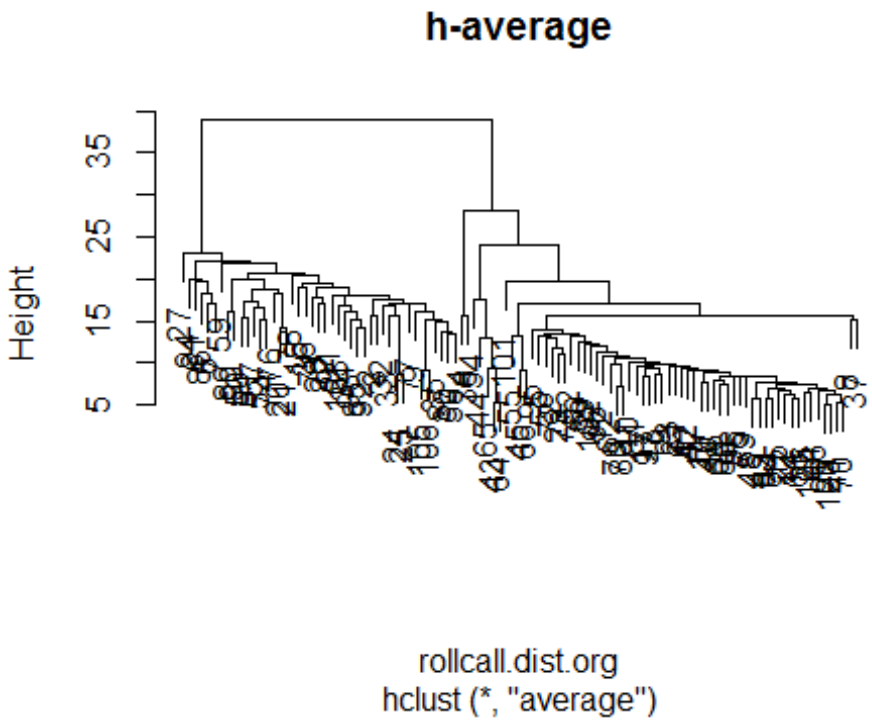

```r
#hclust
h2single<-hclust(rollcall.dist.org,method = "single")
plot(h2single, main = "h-single")
```



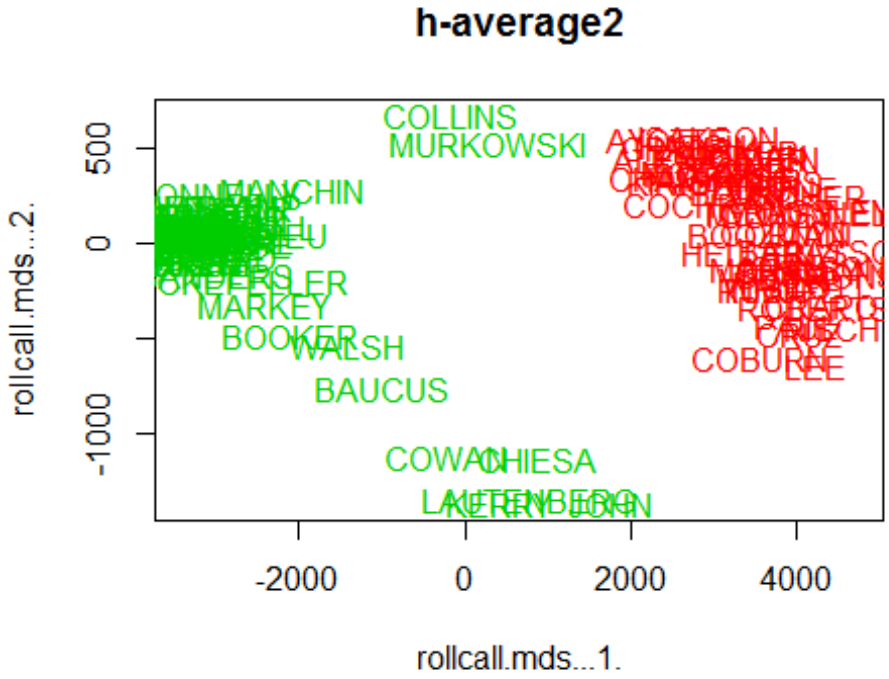
```
h2single2<-cutree(h2single,k=2)
plot(rollcall.mds2, type = 'n',main="h-single2")
text(rollcall.mds2, labels=rollcall.mds$name, col = h2single2+1)
```



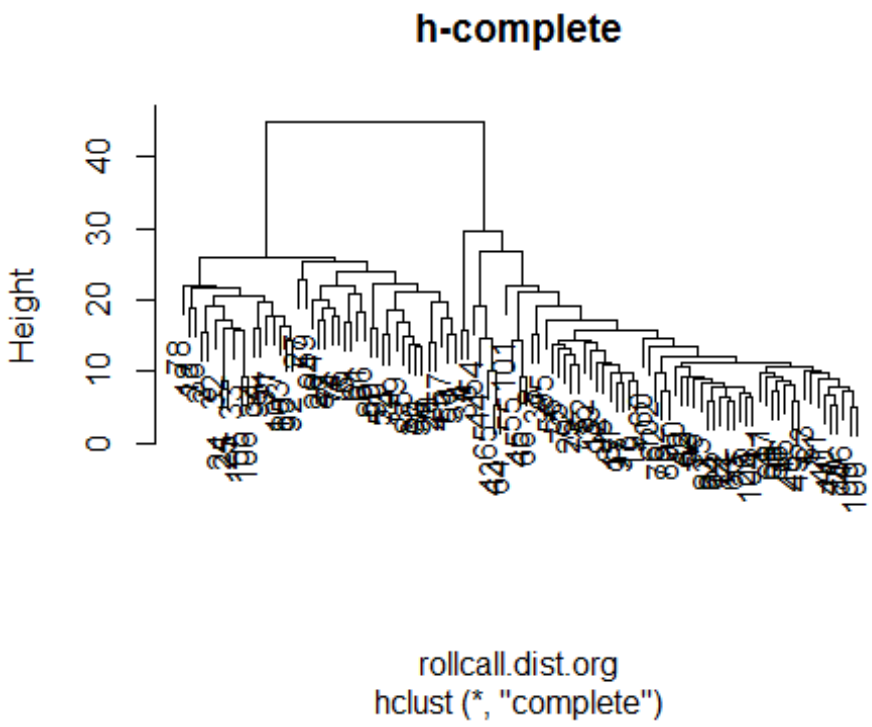

```
h2average<-hclust(rollcall.dist.org,method = "average")
plot(h2average, main = "h-average")
```



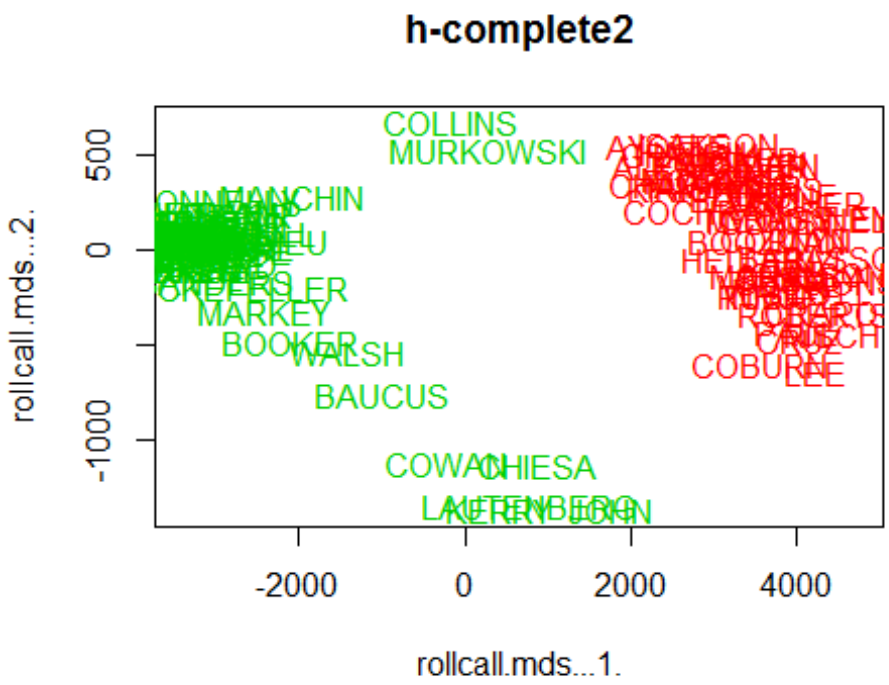

```
h2average2<-cutree(h2average,k=2)
plot(rollcall.mds2, type = 'n', main= "h-average2")
text(rollcall.mds2, labels=rollcall.mds$name, col = h2average2+1)
```



```r
h2complete<-hclust(rollcall.dist.org,method = "complete")
plot(h2complete, main = "h-complete")
```




```r
h2complete2<-cutree(h2complete,k=2)
plot(rollcall.mds2, type = 'n', main="h-complete2")
text(rollcall.mds2, labels=rollcall.mds$name, col = h2complete2+1)
```



##3

```
# k-means:
# COLLINS should be republican but wrongly clustered as democrat.
# LAUTENBERG should be democrats but wrongly clustered as republican.
# KERRY JOHN should be democrats but wrongly clustered as republican.
#
#h-single:
#All the democrats(left-side) are wrongly clustered as Republicans and
COLLINS,MURKOWSKI should be republicans but wrongly clustered as democrat.
#
#h-average and h-complete:
#COLLINS,MURKOWSKI, CHIESA should be republicans but wrongly clustered as
democrats.
```

##4

```
cluster.purity <- function(classes, clusters) {
  sum(apply(table(classes, clusters), 2, max)) / length(clusters)
}

cluster.entropy <- function(classes,clusters) {
  en <- function(x) {
    s = sum(x)
    sum(sapply(x/s, function(p) {if (p) -p*log2(p) else 0} ) )
  }
  M = table(classes, clusters)
```

```
  m = apply(M, 2, en)
  c = colSums(M) / sum(M)
  sum(m*c)
}

classess<-recode(rollcall.mds$party,"200"="1","100"="2")

kmeans_clusters<-as.factor(kmeans2_rollcall$cluster)
h2single_clusers<-as.factor(h2single2)
h2average_clusers<-as.factor(h2average2)
h2complete_clusers<-as.factor(h2complete2)

kmeans_p<-cluster.purity(classess,kmeans_clusters)
h2single_p<-cluster.purity(classess,h2single_clusers)
h2average_p<-cluster.purity(classess,h2average_clusers)
h2complete_p<-cluster.purity(classess,h2complete_clusers)

purity<-c(kmeans_p,h2single_p,h2average_p,h2complete_p)

kmeans_e<-cluster.entropy(classess,kmeans_clusters)
h2single_e<-cluster.entropy(classess,h2single_clusers)
h2average_e<-cluster.entropy(classess,h2average_clusers)
h2complete_e<-cluster.entropy(classess,h2complete_clusers)

entropy<-c(kmeans_e,h2single_e,h2average_e,h2complete_e)

tab<-rbind(purity,entropy)
colnames(tab)<-c('k-means','hclust-single','hclust-average','hclust-
complete')
tab
```

```
##          k-means hclust-single hclust-average hclust-complete
## purity  0.952381     0.5619048      0.9523810       0.9523810
## entropy 0.302099     1.0859032      0.2850529       0.2850529
```

##5

```
# From both the measures and mis-classified members, h-average and h-complete
seems to be most meaningful cluster results with high purity and low entropy
compared to others.
```