```
##########################################################################
## Course: Machine Learning for Economists and Business Analysts
## Topic: Unsupervised Learning - PCA, Clustering
##########################################################################

#setwd("")

# Load data
load("rollcall-votes.Rdata")
load("rollcall-members.Rdata")



###################################################
### Exercise 1: Data Description and Visualization ###
###################################################

##########################################################################
# Task 1
##########################################################################

# Counts of Democrats, Republicans and one special politician
table(members$party)

# Shares of Democrats, Republicans and one special politician
round(table(members$party)/nrow(members),3)

##########################################################################
# Task 2
##########################################################################

# Count missing votings for each politician and plot the counts
missings <- rowSums(votes==0)

# No. politicians who always voted
sum(missings == 0)

# Shares of missing votings
s_missings <- missings/ncol(votes)


# Histogram with 100 bins
hist(s_missings, breaks = 100)

##########################################################################
# Task 3
##########################################################################

# Counts - yes and nos
yeas <- rowSums(votes==1)
nays <- rowSums(votes==-1)

# Plots - Party
```

```
plot(yeas, nays, col = members$party)
legend('topleft', legend = levels(members$party), col = 1:3,  pch = 1)

################################################################################


#######################################################
### Exercise 2: Principal Component Analysis (PCA) ###
#######################################################

################################################################################
# Task 1
################################################################################

# PCA
pr.out = prcomp(votes , center = TRUE, scale = TRUE)

# No of principal components
dim(pr.out$rotation)[2]

################################################################################
# Task 2
################################################################################

# variance explained by each component
pr.var = pr.out$sdev^2

# Proportion of variance explained
pve=pr.var/sum(pr.var)

# Print first 10 PC
pve[1:10]

# Plot the first 10 PC
barplot(pve[1:10], xlab=" Principal Component ", ylab=" Proportion of Variance Explained ",
ylim=c(0,1))
barplot(cumsum(pve[1:10]), xlab=" Principal Component ", ylab ="Cumulative Proportion of Variance
Explained ", ylim=c(0,1))

################################################################################
# Task 3
################################################################################

# Plot the first two principal components color the party membership
plot(pr.out$x[,1], pr.out$x[,2], xlab = "PC1", ylab = "PC2", col = members$party, main = "Top two PC
directions")
legend('bottomright', legend = levels(members$party), col = 1:3,  pch = 1)

################################################################################
# Task 4
################################################################################

## Far right (very conservative)
```

```r
head(sort(pr.out$x[,1]))

## Far left (very liberal)
head(sort(pr.out$x[,1], decreasing=TRUE))

###################################################################################
# Task 5
###################################################################################

# PC 2
head(sort(pr.out$x[,2]))
# No clear pattern based on party and state information

# Look at the largest loadings in PC2 to discern an interpretation.
loadings <- pr.out$rotation
loadings[order(abs(loadings[,2]), decreasing=TRUE)[1:5],2]

# Analyze voting behavior
table(votes[,1146])
table(votes[,658])
table(votes[,1090])

# Either everyone voted "yea" or missed the voting.
# These votes all correspond to near-unanimous symbolic action.

# Mystery Solved: the second PC is just attendance!
head(sort(rowSums(votes==0), decreasing=TRUE))

###################################################################################


########################################
### Exercise 3: k-means Clustering ###
########################################

###################################################################################
# Task 1
###################################################################################

set.seed(11122019)

# K-means clustering with 2 clusters
km.out = kmeans(votes, 2, nstart = 20)
km.out$cluster

# Tabulate party vs cluster
table(members$party, km.out$cluster)

###################################################################################
# Task 2
###################################################################################

# How to analyze the optimal number of clusters
```

```
sse <- c()
sse[1] <- Inf

for (ind_cl in c(2:20)) {

  set.seed(3)
  km.out = kmeans (votes, ind_cl, nstart = 20)
  sse[ind_cl] = km.out$tot.withinss

}

plot(sse)
# Optimum 4-5 clusters

###################################################################################
# Task 3
###################################################################################
# Plot the 5 clusters on the PC components graph

set.seed(3)
km.out = kmeans (votes, 5, nstart = 20)

# Plot the first two principal components color the party membership
plot(pr.out$x[,1], pr.out$x[,2], xlab = "PC1", ylab = "PC2", col = km.out$cluster, main = "Top two PC directions with 5 clusters")
legend('bottomright', legend = c("Cluster 1", "Cluster 2", "Cluster 3", "Cluster 4", "Cluster 5"), col = 1:5,  pch = 1)

###################################################################################
# Task 4
###################################################################################

# Analyzing how the number of starts work
set.seed (3)
km.out = kmeans (votes,6, nstart = 1)
km.out$tot.withinss

km.out =kmeans (votes,6, nstart = 20)
km.out$tot.withinss
```