NBAPlayersClustering.R

danielpeslherbe

2020-10-03

```
##NBA Players Clustering

options(warn = -1)
##install.packages('tidyverse')
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.1
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

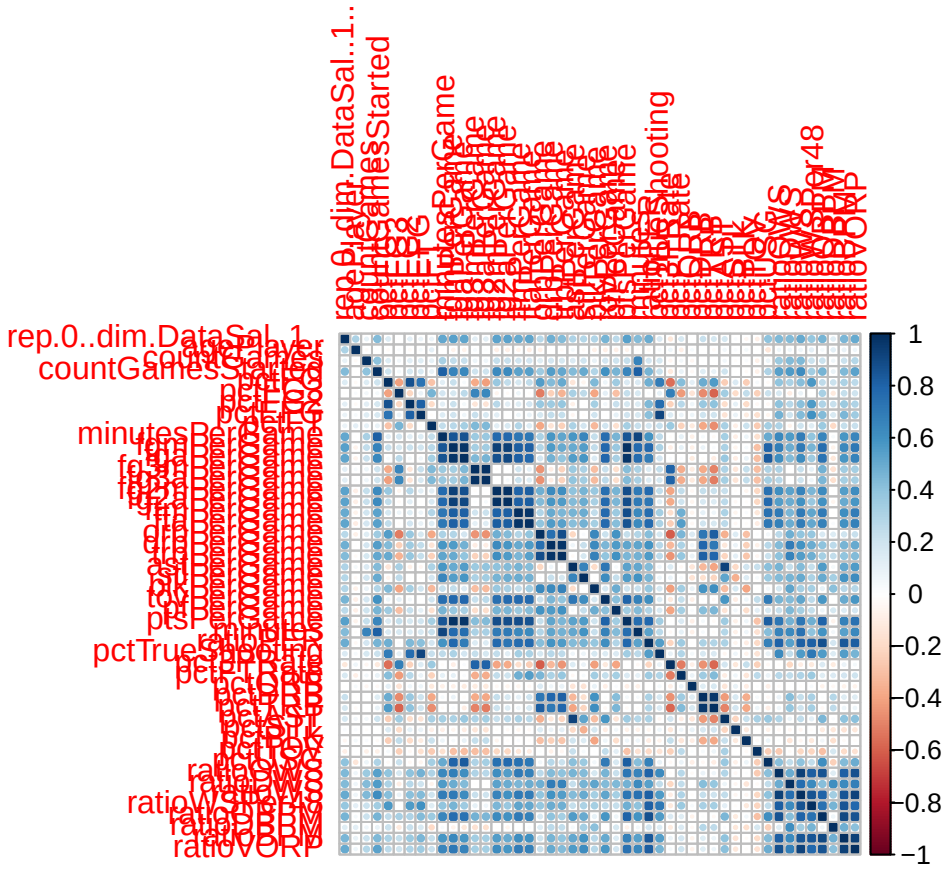
```
##install.packages('xlsx')
library(xlsx)
##install.packages('openxlsx')
library(openxlsx)
```

```
##
## Attaching package: 'openxlsx'
## The following objects are masked from 'package:xlsx':
##
##     createWorkbook, loadWorkbook, read.xlsx, saveWorkbook, write.xlsx
```

```
##Import Data
DataMaster <- read.csv('~/Library/Mobile Documents/com~apple~CloudDocs/NBA Players Project/Dataset1990-9
##Remove data point indices
DataMaster <- DataMaster[,-1]
##Note that for this clustering, we must remove players with very small playing time
##or small amounts of games played, since their impact is minimal
DataMasterFiltered <- DataMaster %>% filter(countGames >= 41, minutesPerGame >= 12)
##Take only data used for cluster analysis
DataCluster <- DataMasterFiltered[,-c(1:3)]

##install.packages('cluster')
library(cluster)
##install.packages('corrplot')
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
correlation.matrix <- cor(DataCluster)
corrplot(correlation.matrix)
```




```
##We notice many of the variables have significant correlation;
##but this is to be expected, as you usually want to play high minutes
##if you are going to score a lot of points, and the negative correlation
##between 3-point shot rate and rebounding percentage is explained by the
##fact that a 3-point shot implies distance from the rim, thus from most
##rebounds (unless it is a long rebound, but the goal of this project is
##not shot mechanics and ball physics.)

##To reduce the number of variables used, we use Principal Component
##Analysis; the first step before applying this is to normalize the data
##to reduce scaling effects (points per game are measured on a scale of 0 to
##the maximum ppg in a season, vs a scale of 0 to 1 for shooting percentage).

##Note that to get a better idea of player value, we must take our
##statistics relative to league average (this helps mitigate the effects
##of changes in gameplay throughout the years, i.e. classifying the best
##scorers or efficient shooters relative to league average that year, and then
##scaling every year together as well)
##Note that we cluster players based on diret stats only (advanced metrics related
##to game winning impact i.e. WS, VROP, etc... will be used in another cluster
##analysis to determine which players have a winning impact)
temp <- c()
DataClusterStatsNorm <- c()
```

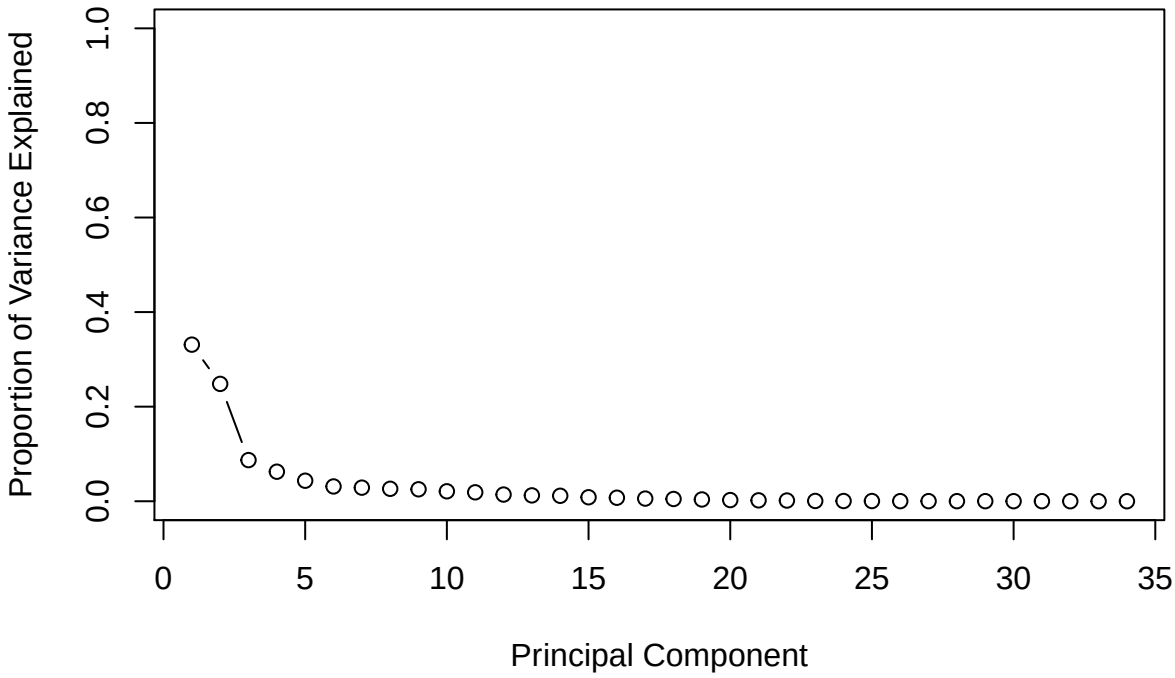
```
centers.stats <- c()
stddevs.stats <- c()
for (i in 1:length(unique(DataMasterFiltered$Season))) {
  for (j in 1:dim(DataMasterFiltered)[1]) {
    if(DataMasterFiltered$Season[j] == unique(DataMasterFiltered$Season)[i]){
      temp <- rbind(temp, DataCluster[j,-c(1:4,10,28,41:48)])
    }
  }
  DataClusterStatsNorm <- rbind(DataClusterStatsNorm, scale(temp))
  centers.stats <- rbind(centers.stats, colMeans(temp))
  stddevs.stats <- rbind(stddevs.stats, apply(temp, 2, sd))
  temp <- c()
}
##use colMeans to get overall average
centers.general.stats <- colMeans(DataClusterStatsNorm)
##likewise, we record the overall standard deviations for each variable
stddevs.general.stats <- apply(DataClusterStatsNorm, 2, sd)
##scale again
DataClusterStatsNorm <- scale(DataClusterStatsNorm)
DataClusterStatsNorm <- as.data.frame(DataClusterStatsNorm)

##use prcomp() fpr PCA
pca <- prcomp(DataClusterStatsNorm)
##Store variance in separate vector
pcavar <- (pca$sdev)^2
pcavar
```

```
##  [1] 1.125934e+01 8.436429e+00 2.958553e+00 2.123288e+00 1.476169e+00
##  [6] 1.063366e+00 9.772786e-01 8.941832e-01 8.592034e-01 7.139771e-01
## [11] 6.419552e-01 4.809777e-01 4.210975e-01 3.925481e-01 2.863964e-01
## [16] 2.520373e-01 1.913101e-01 1.629452e-01 1.269487e-01 8.359665e-02
## [21] 6.498222e-02 5.083911e-02 2.102722e-02 1.617447e-02 1.464714e-02
## [26] 7.566295e-03 6.083069e-03 5.052313e-03 4.424352e-03 3.890123e-03
## [31] 2.961119e-03 3.728875e-04 2.195979e-04 1.630707e-04
```

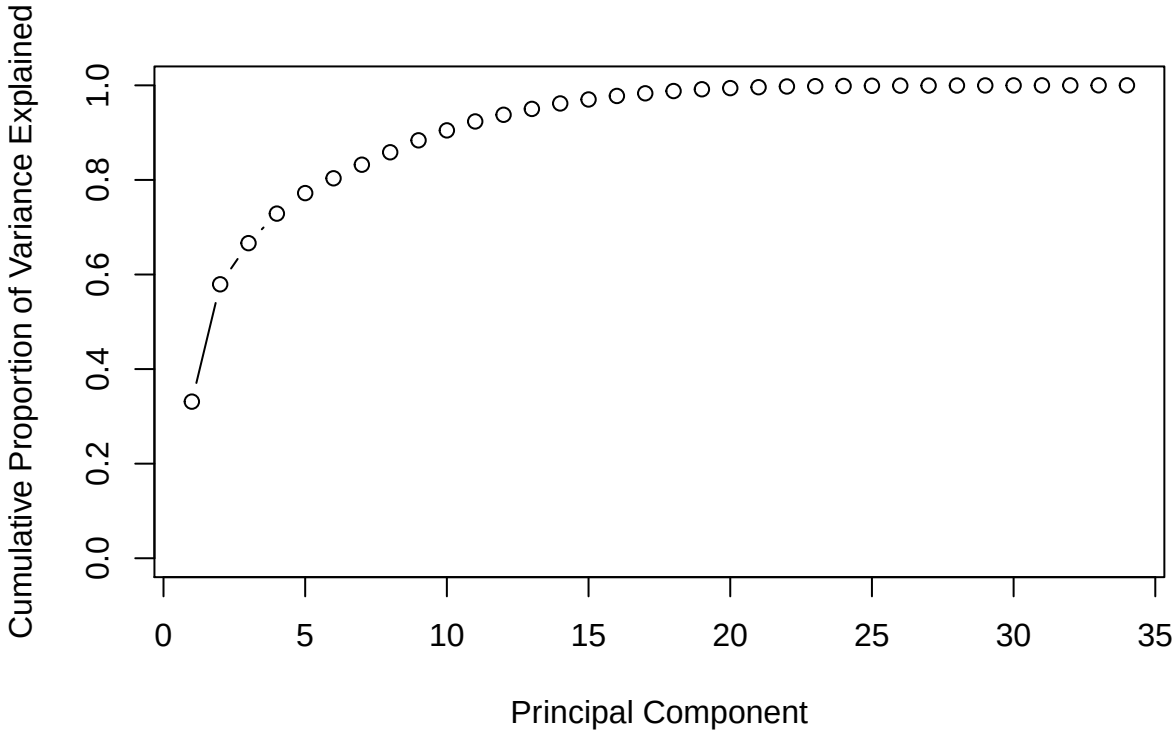
```
##Store variance explained by each Principal Component
pcavarexpl <- pcavar/sum(pcavar)
pcavarexpl
```

```
##  [1] 3.311570e-01 2.481303e-01 8.701627e-02 6.244965e-02 4.341674e-02
##  [6] 3.127548e-02 2.874349e-02 2.629950e-02 2.527069e-02 2.099933e-02
## [11] 1.888103e-02 1.414640e-02 1.238522e-02 1.154553e-02 8.423422e-03
## [16] 7.412862e-03 5.626768e-03 4.792506e-03 3.733786e-03 2.458725e-03
## [21] 1.911242e-03 1.495268e-03 6.184475e-04 4.757198e-04 4.307982e-04
## [26] 2.225381e-04 1.789138e-04 1.485974e-04 1.301280e-04 1.144154e-04
## [31] 8.709172e-05 1.096728e-05 6.458761e-06 4.796197e-06
```

```
##Plot variance and cumulative variance explained by Principal Component
plot(pcavarexpl, xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1),type=
```




```
plot(cumsum(pcavarexpl), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained"
```

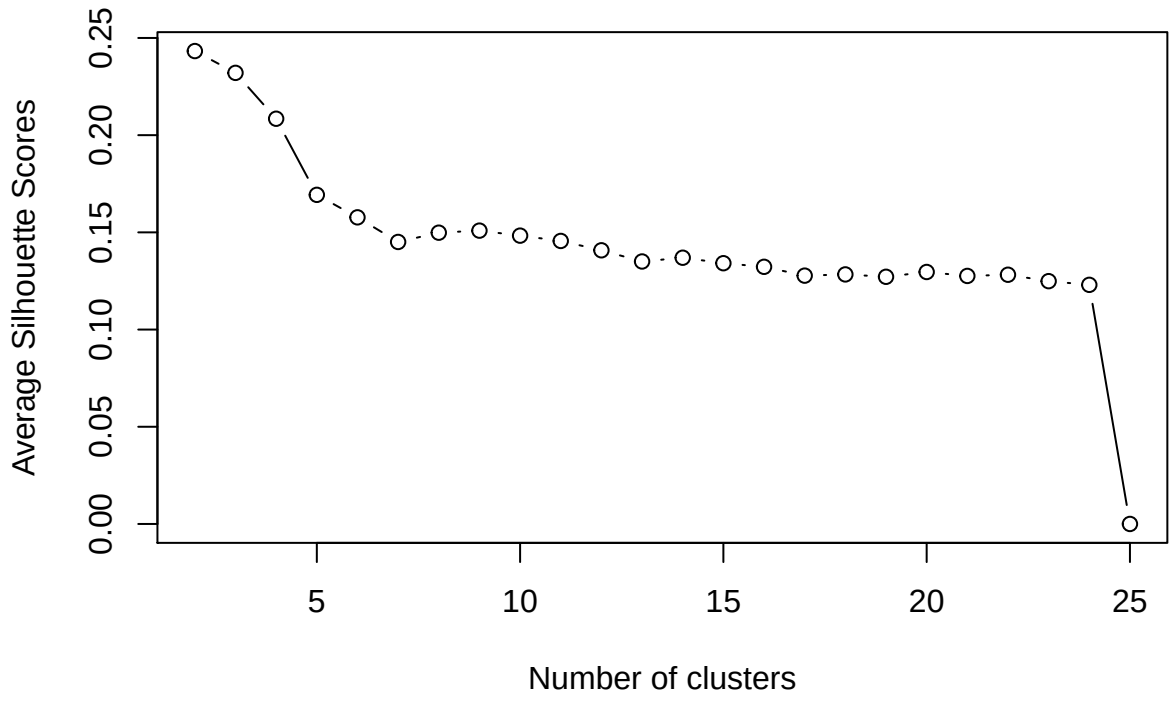



```
##Create vector of cumulative variance explained
pcatotvarexpl <- cumsum(pcavarexpl)
##Set a variance threshold
var.accept <- 0.90
##Compute the minimum number of Principal Components to reach variance threshold
princip.comp.used <- min(which(pcatotvarexpl >= var.accept))
##Store PCA for the minimum number of Principal Components
DataPCA <- pca$x[,1:princip.comp.used]

##install.packages('factoextra')
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
##install.packages('NbClust')
library(NbClust)

##The following methods have proved too narrow to cluster the dataset into relevant categories
##Elbow Method
##fviz_nbclust(DataPCA, kmeans, method = "wss") +
##  geom_vline(xintercept = 4, linetype = 2) +
##  labs(subtitle = "Elbow method")

# Silhouette method
##fviz_nbclust(DataPCA, kmeans, method = "silhouette") +
##  labs(subtitle = "Silhouette method")
```

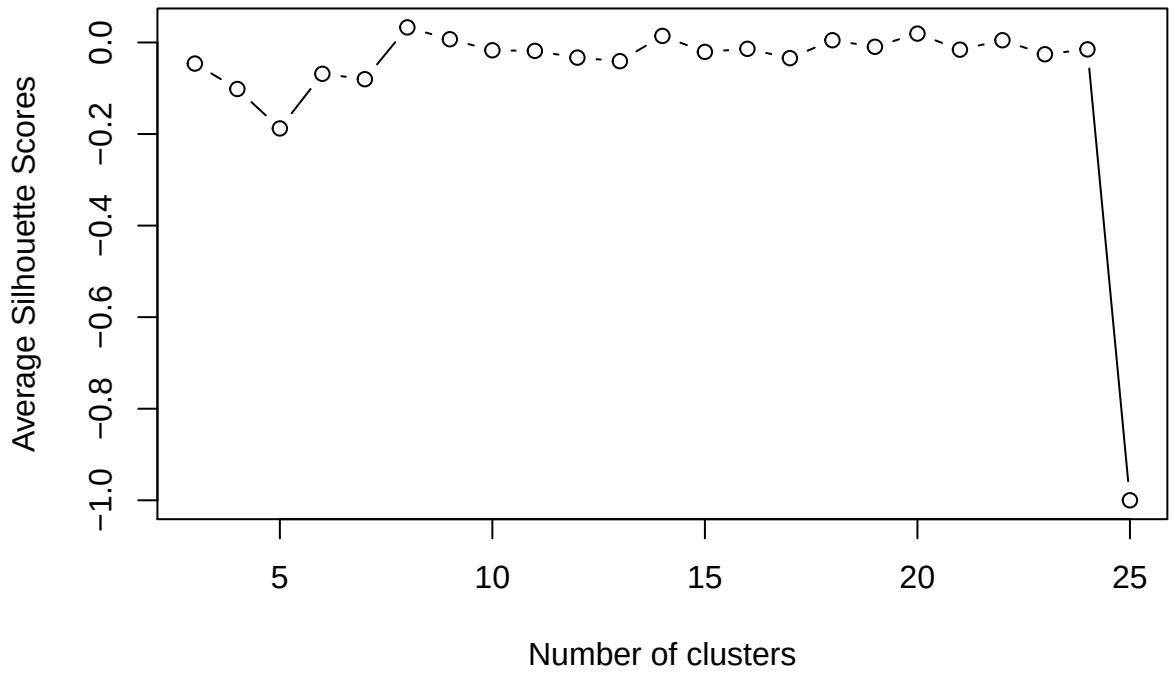
```
# Gap statistic
##set.seed(4)
##fviz_nbclust(DataPCA, kmeans, nstart = 25, method = "gap_stat", k.max = 25, nboot = 500) +
## labs(subtitle = "Gap statistic method")

##NbClust method
##nbclust_out <- NbClust(data = DataPCA,  distance = "euclidean",  min.nc = 5, max.nc = 25, method = "k

##Thus, let us re-use silhouettescore() function code from my Pokemon K Means Project (with improvement
silhouettescore <- function(k){
  km <- kmeans(DataPCA, centers = k, nstart = 250)
  ss <- silhouette(km$cluster, dist(DataPCA))
  mean(ss[,3])
}
k <- 2:25
avgss <- c(rep(0,24))
for (i in min(k):length(k)) {
  avgss[i-1] <- silhouettescore(i)
}
plot(k, type = 'b', avgss, xlab = 'Number of clusters', ylab = 'Average Silhouette Scores')
```




```
optk <- which.max(avgss)+1
optk
```

```
## [1] 2
```

```
ssimprov <- c(rep(0,23))
for (j in 1:length(ssimprov)) {
  ssimprov[j] <- avgss[j+1]/avgss[j] -1
}
plot(k[2:24], type = 'b', ssimprov, xlab = 'Number of clusters', ylab = 'Average Silhouette Scores')
```




```
optstatk <- which.max(ssimprov)+2
optstatk
```

```
## [1] 8
```

```
ClusterStatResult <- kmeans(DataPCA, centers = optk, nstart = 500)
DataClustered <- cbind(DataMasterFiltered, ClusterStatResult$cluster)

##Let us reapply this on our advanced metrics

temp <- c()
DataClusterAdvNorm <- c()
centers.adv <- c()
stddevs.adv <- c()
for (i in 1:length(unique(DataMasterFiltered$Season))) {
  for (j in 1:dim(DataMasterFiltered)[1]) {
    if(DataMasterFiltered$Season[j] == unique(DataMasterFiltered$Season)[i]){
      temp <- rbind(temp, DataCluster[j,c(41:48)])
    }
  }
  DataClusterAdvNorm <- rbind(DataClusterAdvNorm, scale(temp))
```

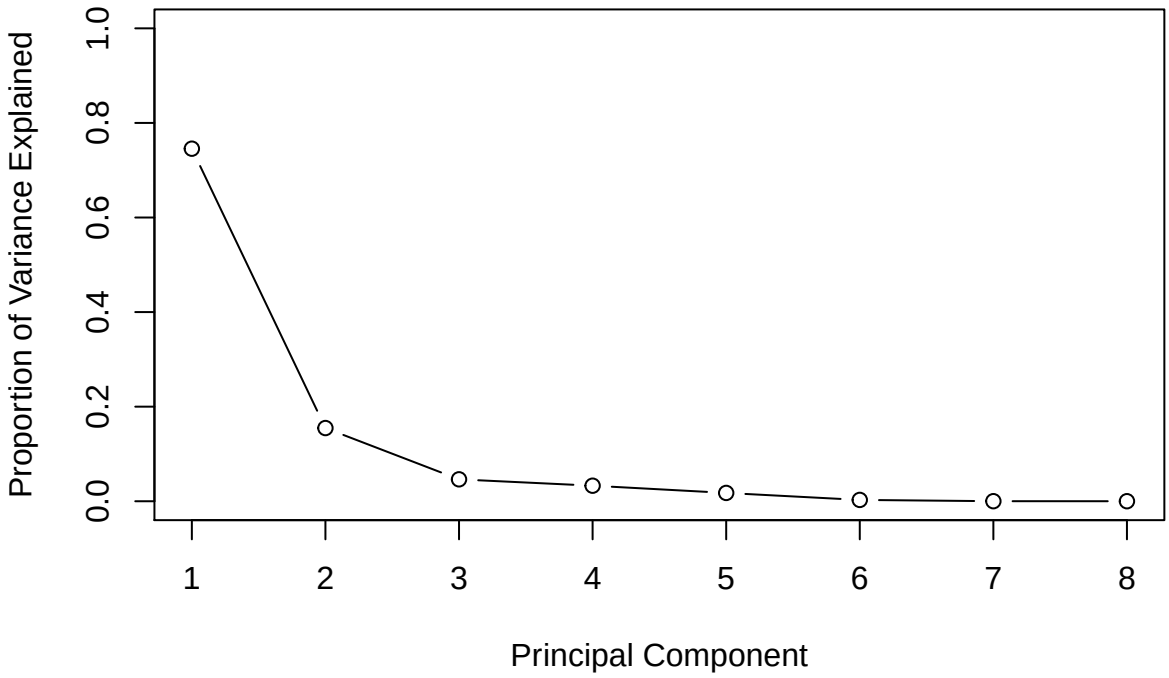
```
  centers.adv <- rbind(centers.adv, colMeans(temp))
  stddevs.adv <- rbind(stddevs.adv, apply(temp, 2, sd))
  temp <- c()
}
##use colMeans to get overall average
centers.general.adv <- colMeans(DataClusterAdvNorm)
##likewise, we record the overall standard deviations for each variable
stddevs.general.adv <- apply(DataClusterAdvNorm, 2, sd)
##scale again
DataClusterAdvNorm <- scale(DataClusterAdvNorm)
DataClusterAdvNorm <- as.data.frame(DataClusterAdvNorm)

##use prcomp() fpr PCA
pca <- prcomp(DataClusterAdvNorm)
##Store variance in separate vector
pcavar <- (pca$sdev)^2
pcavar
```

```
## [1] 5.9642724722 1.2381314684 0.3704556439 0.2630799758 0.1405032828
## [6] 0.0222912903 0.0008427187 0.0004231481
```

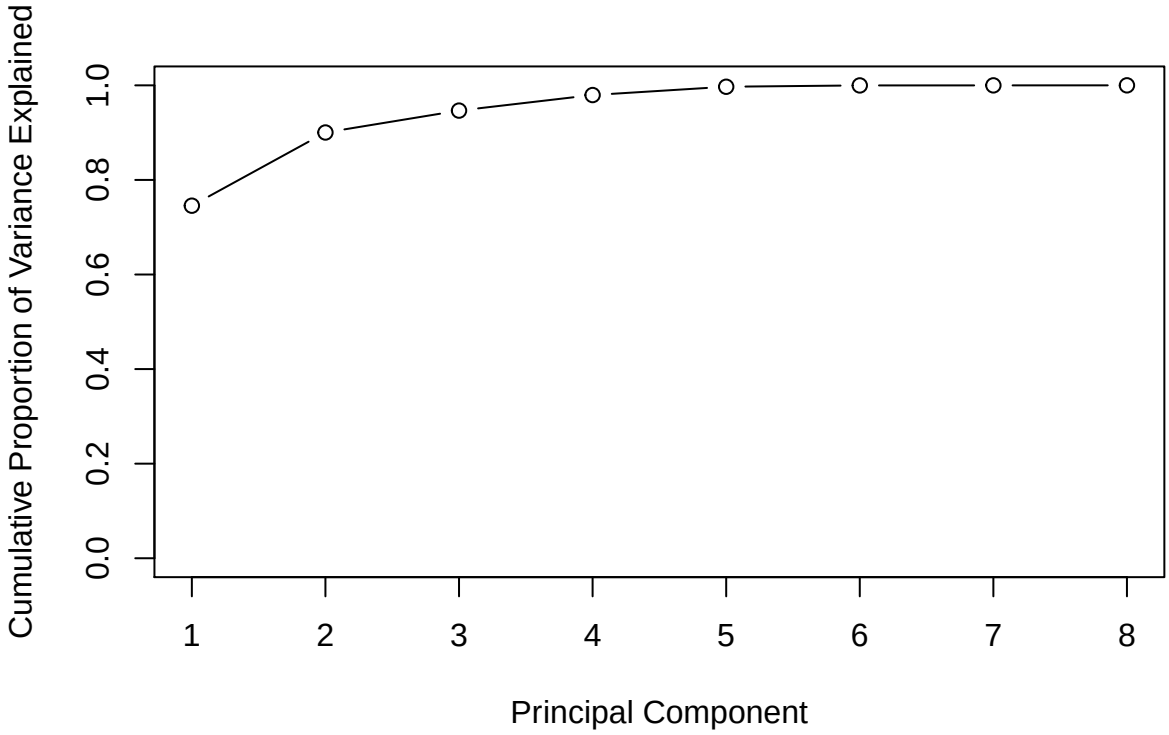
```
##Store variance explained by each Principal Component
pcavarexpl <- pcavar/sum(pcavar)
pcavarexpl
```

```
## [1] 7.455341e-01 1.547664e-01 4.630696e-02 3.288500e-02 1.756291e-02
## [6] 2.786411e-03 1.053398e-04 5.289351e-05
```

```
##Plot variance and cumulative variance explained by Principal Component
plot(pcavarexpl, xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1),type=
```

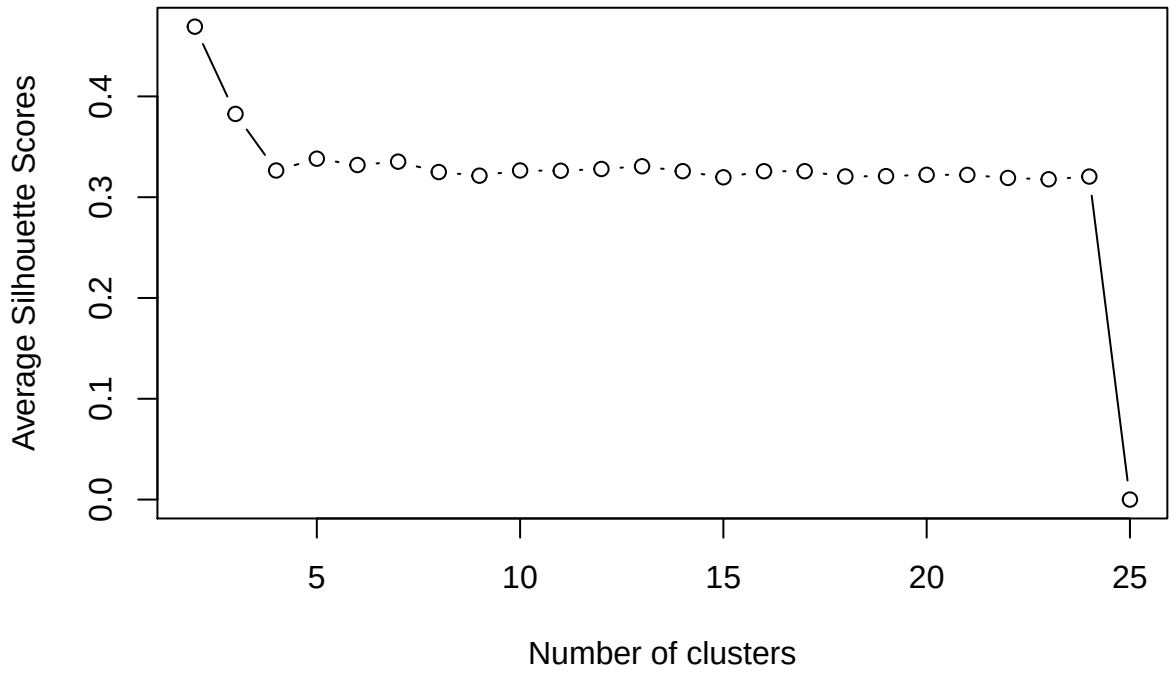



```
plot(cumsum(pcavarexpl), xlab="Principal Component", ylab="Cumulative Proportion of Variance Explained"
```




```
##Create vector of cumulative variance explained
pcatotvarexpl <- cumsum(pcavarexpl)
##Set a variance threshold
var.accept <- 0.90
##Compute the minimum number of Principal Components to reach variance threshold
princip.comp.used <- min(which(pcatotvarexpl >= var.accept))
##Store PCA for the minimum number of Principal Components
DataPCA <- pca$x[,1:princip.comp.used]

##Elbow Method
##fviz_nbclust(DataPCA, kmeans, method = "wss") +
##  geom_vline(xintercept = 4, linetype = 2) +
##  labs(subtitle = "Elbow method")

# Silhouette method
##fviz_nbclust(DataPCA, kmeans, method = "silhouette") +
##  labs(subtitle = "Silhouette method")

# Gap statistic
##set.seed(4)
##fviz_nbclust(DataPCA, kmeans, nstart = 25, method = "gap_stat", k.max = 25, nboot = 500) +
## labs(subtitle = "Gap statistic method")

##NbClust method
##nbclust_out <- NbClust(data = DataPCA,  distance = "euclidean",  min.nc = 5, max.nc = 25, method = "k
```

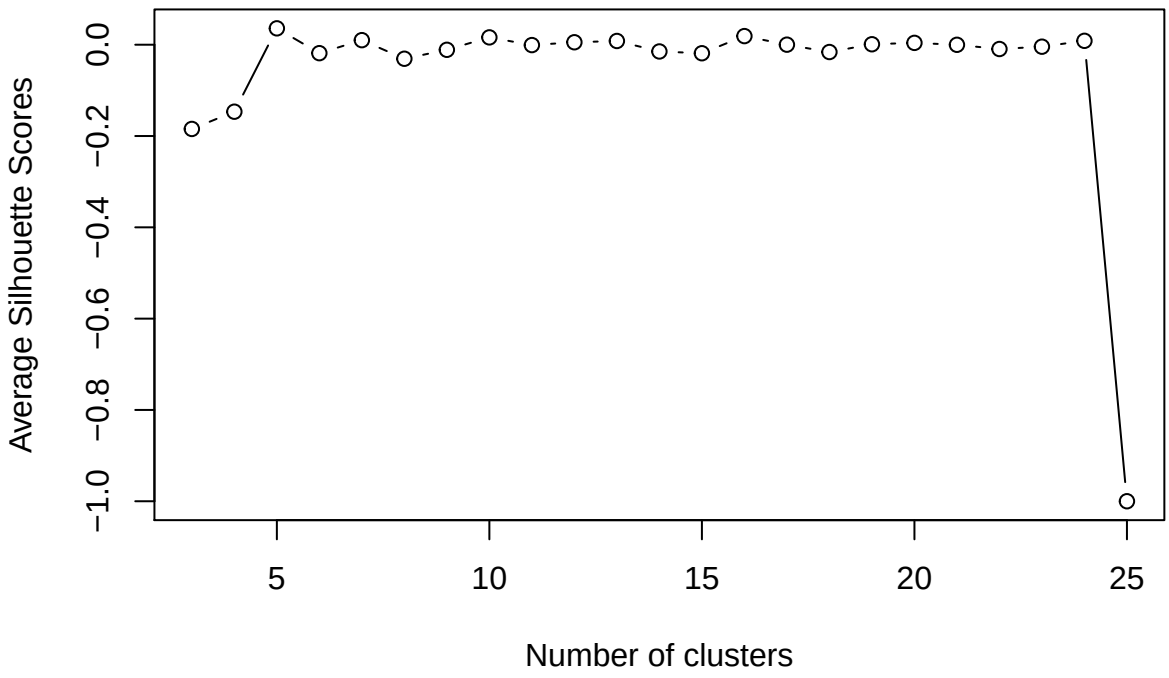
```
silhouettescore <- function(k){
  km <- kmeans(DataPCA, centers = k, nstart = 250)
  ss <- silhouette(km$cluster, dist(DataPCA))
  mean(ss[,3])
}
k <- 2:25
avgss <- c(rep(0,24))
for (i in min(k):length(k)) {
  avgss[i-1] <- silhouettescore(i)
}
plot(seq(2,max(k)), type = 'b', avgss, xlab = 'Number of clusters', ylab = 'Average Silhouette Scores')
```




```
optk <- which.max(avgss)+1
optk
```

```
## [1] 2
```

```
ssimprov <- c(rep(0,23))
for (j in 1:length(ssimprov)) {
  ssimprov[j] <- avgss[j+1]/avgss[j] -1
}
plot(seq(3,max(k)), type = 'b', ssimprov, xlab = 'Number of clusters', ylab = 'Average Silhouette Scores
```




```
optadvk <- which.max(ssimprov)+2
optadvk
```

```
## [1] 5
```

```
ClusterAdvResult <- kmeans(DataPCA, centers = optk, nstart = 500)
DataClustered <- cbind(DataClustered, ClusterAdvResult$cluster)
DataClustered <- DataClustered %>% rename(StatsCluster = `ClusterStatResult$cluster`,
         AdvCluster = `ClusterAdvResult$cluster`,
         SalaryPercentage = rep.0..dim.DataSal..1..)

##note that we have opstatk * optadvk = 8 * 5 = 40 group possibilities now
ClusterList <- vector(mode = 'list', length = optstatk*optadvk)

for (j in 1:optadvk) {
  for (k in 1:optstatk) {
    for (i in 1:dim(DataClustered)[1]) {
      l <- (j-1)*8 +k
      if(DataClustered$StatsCluster[i] == k && DataClustered$AdvCluster[i] == j){
      ClusterList[[l]] <- rbind(ClusterList[[l]], DataClustered[i,])
     }
   }
 }
}

ClusterCenters <- c()
for(i in 1:length(ClusterList)) {
```

```
  if(!is_empty(ClusterList[[i]] == TRUE)){
  ClusterCenters <- rbind(ClusterCenters, colMeans(ClusterList[[i]][-c(1,2)]))
  }
}

##Export DataClustered into Excel according to Stats Clusters
StatsCluster <- createWorkbook()
addWorksheet(StatsCluster, 'Full Set')
addWorksheet(StatsCluster, '1st Cluster')
addWorksheet(StatsCluster, '2nd Cluster')
addWorksheet(StatsCluster, '3rd Cluster')
addWorksheet(StatsCluster, '4th Cluster')
addWorksheet(StatsCluster, '5th Cluster')
addWorksheet(StatsCluster, '6th Cluster')
addWorksheet(StatsCluster, '7th Cluster')
addWorksheet(StatsCluster, '8th Cluster')
writeData(StatsCluster, sheet = 'Full Set', x = DataClustered)
writeData(StatsCluster, sheet = '1st Cluster', x = rbind(ClusterList[[1]], ClusterList[[9]], ClusterLis
writeData(StatsCluster, sheet = '2nd Cluster', x = rbind(ClusterList[[10]], ClusterList[[18]], ClusterL
writeData(StatsCluster, sheet = '3rd Cluster', x = rbind(ClusterList[[3]], ClusterList[[11]], ClusterLis
writeData(StatsCluster, sheet = '4th Cluster', x = rbind(ClusterList[[4]], ClusterList[[12]], ClusterLis
writeData(StatsCluster, sheet = '5th Cluster', x = rbind(ClusterList[[5]], ClusterList[[13]], ClusterLis
writeData(StatsCluster, sheet = '6th Cluster', x = rbind(ClusterList[[6]], ClusterList[[14]], ClusterLis
writeData(StatsCluster, sheet = '7th Cluster', x = rbind(ClusterList[[7]], ClusterList[[15]], ClusterLis
writeData(StatsCluster, sheet = '8th Cluster', x = rbind(ClusterList[[16]], ClusterList[[24]], ClusterL
saveWorkbook(StatsCluster, file = 'NBAStatsClustering.xlsx')

##Export DataClustered into Excel according to Adv Clusters
AdvCluster <- createWorkbook()
addWorksheet(AdvCluster, 'Full Set')
addWorksheet(AdvCluster, '1st Cluster')
addWorksheet(AdvCluster, '2nd Cluster')
addWorksheet(AdvCluster, '3rd Cluster')
addWorksheet(AdvCluster, '4th Cluster')
addWorksheet(AdvCluster, '5th Cluster')
writeData(AdvCluster, sheet = 'Full Set', x = DataClustered)
writeData(AdvCluster, sheet = '1st Cluster', x = rbind(ClusterList[[1]], ClusterList[[3]], ClusterList[
writeData(AdvCluster, sheet = '2nd Cluster', x = rbind(ClusterList[[9]], ClusterList[[10]], ClusterList
writeData(AdvCluster, sheet = '3rd Cluster', x = rbind(ClusterList[[18]], ClusterList[[19]], ClusterLis
writeData(AdvCluster, sheet = '4th Cluster', x = rbind(ClusterList[[25]], ClusterList[[26]], ClusterLis
writeData(AdvCluster, sheet = '5th Cluster', x = rbind(ClusterList[[33]], ClusterList[[34]], ClusterLis
saveWorkbook(AdvCluster, file = 'NBAAdvClustering.xlsx')


##Export DataClustered into Excel according to Double Clustering
Cluster <- createWorkbook()
addWorksheet(Cluster, 'Full Set')
addWorksheet(Cluster, '1st Cluster')
addWorksheet(Cluster, '3rd Cluster')
addWorksheet(Cluster, '4th Cluster')
addWorksheet(Cluster, '5th Cluster')
addWorksheet(Cluster, '6th Cluster')
addWorksheet(Cluster, '7th Cluster')
```

```
addWorksheet(Cluster, '9th Cluster')
addWorksheet(Cluster, '10th Cluster')
addWorksheet(Cluster, '11th Cluster')
addWorksheet(Cluster, '12th Cluster')
addWorksheet(Cluster, '13th Cluster')
addWorksheet(Cluster, '14th Cluster')
addWorksheet(Cluster, '15th Cluster')
addWorksheet(Cluster, '16th Cluster')
addWorksheet(Cluster, '17th Cluster')
addWorksheet(Cluster, '18th Cluster')
addWorksheet(Cluster, '19th Cluster')
addWorksheet(Cluster, '20th Cluster')
addWorksheet(Cluster, '21th Cluster')
addWorksheet(Cluster, '22nd Cluster')
addWorksheet(Cluster, '23rd Cluster')
addWorksheet(Cluster, '24th Cluster')
addWorksheet(Cluster, '25th Cluster')
addWorksheet(Cluster, '26th Cluster')
addWorksheet(Cluster, '27th Cluster')
addWorksheet(Cluster, '28th Cluster')
addWorksheet(Cluster, '29th Cluster')
addWorksheet(Cluster, '30th Cluster')
addWorksheet(Cluster, '31st Cluster')
addWorksheet(Cluster, '32nd Cluster')
addWorksheet(Cluster, '33rd Cluster')
addWorksheet(Cluster, '34th Cluster')
addWorksheet(Cluster, '35th Cluster')
addWorksheet(Cluster, '36th Cluster')
addWorksheet(Cluster, '37th Cluster')
addWorksheet(Cluster, '38th Cluster')
addWorksheet(Cluster, '39th Cluster')
addWorksheet(Cluster, '40th Cluster')
writeData(Cluster, sheet = 'Full Set', x = DataClustered)
writeData(Cluster, sheet = '1st Cluster', x = ClusterList[[1]])
writeData(Cluster, sheet = '3rd Cluster', x = ClusterList[[3]])
writeData(Cluster, sheet = '4th Cluster', x = ClusterList[[4]])
writeData(Cluster, sheet = '5th Cluster', x = ClusterList[[5]])
writeData(Cluster, sheet = '6th Cluster', x = ClusterList[[6]])
writeData(Cluster, sheet = '7th Cluster', x = ClusterList[[7]])
writeData(Cluster, sheet = '9th Cluster', x = ClusterList[[9]])
writeData(Cluster, sheet = '10th Cluster', x = ClusterList[[10]])
writeData(Cluster, sheet = '11th Cluster', x = ClusterList[[11]])
writeData(Cluster, sheet = '12th Cluster', x = ClusterList[[12]])
writeData(Cluster, sheet = '13th Cluster', x = ClusterList[[13]])
writeData(Cluster, sheet = '14th Cluster', x = ClusterList[[14]])
writeData(Cluster, sheet = '15th Cluster', x = ClusterList[[15]])
writeData(Cluster, sheet = '16th Cluster', x = ClusterList[[16]])
writeData(Cluster, sheet = '17th Cluster', x = ClusterList[[17]])
writeData(Cluster, sheet = '18th Cluster', x = ClusterList[[18]])
writeData(Cluster, sheet = '19th Cluster', x = ClusterList[[19]])
writeData(Cluster, sheet = '20th Cluster', x = ClusterList[[20]])
writeData(Cluster, sheet = '21th Cluster', x = ClusterList[[21]])
writeData(Cluster, sheet = '22nd Cluster', x = ClusterList[[22]])
```

```
writeData(Cluster, sheet = '23rd Cluster', x = ClusterList[[23]])
writeData(Cluster, sheet = '24th Cluster', x = ClusterList[[24]])
writeData(Cluster, sheet = '25th Cluster', x = ClusterList[[25]])
writeData(Cluster, sheet = '26th Cluster', x = ClusterList[[26]])
writeData(Cluster, sheet = '27th Cluster', x = ClusterList[[27]])
writeData(Cluster, sheet = '28th Cluster', x = ClusterList[[28]])
writeData(Cluster, sheet = '29th Cluster', x = ClusterList[[29]])
writeData(Cluster, sheet = '30th Cluster', x = ClusterList[[30]])
writeData(Cluster, sheet = '31st Cluster', x = ClusterList[[31]])
writeData(Cluster, sheet = '32nd Cluster', x = ClusterList[[32]])
writeData(Cluster, sheet = '33rd Cluster', x = ClusterList[[33]])
writeData(Cluster, sheet = '34th Cluster', x = ClusterList[[34]])
writeData(Cluster, sheet = '35th Cluster', x = ClusterList[[35]])
writeData(Cluster, sheet = '36th Cluster', x = ClusterList[[36]])
writeData(Cluster, sheet = '37th Cluster', x = ClusterList[[37]])
writeData(Cluster, sheet = '38th Cluster', x = ClusterList[[38]])
writeData(Cluster, sheet = '39th Cluster', x = ClusterList[[39]])
writeData(Cluster, sheet = '40th Cluster', x = ClusterList[[40]])
saveWorkbook(Cluster, file = 'NBAClustering.xlsx')
```