Case_Study_3

Chinenye Chinwego

2022-05-04

0 Introduction and Motivation

In this case study, I will make an R Shiny application from the analysis of certain data I obtained from datasetsearch.research.google.com. The shiny application will help in better visualizing my analysis and results as well as varying parameters and features and seeing the effect on the overall result obtained.

1 Selecting the Data Set

I will be working with data related to rare earth elements because this is related to my research work as a graduate student. This data set of information is from a dataset on Trends in the Rare Earth Element Content of U.S. - Based Coal Combustion of U.S. Fly Ashes of varied geological origin.

I started by installing the important packages for this project and imported the dataset.

```
library(dplyr)
```

```
##
## 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(tidyr)
library(readxl)
```

2 Selecting an Algorithm;

I selected clustering as the algorithm suitable for my data set.

3 Mathematical/Statistical Details of the Algorithm

The objective was to identify subgroups (or clusters) in my data set. In the kmeans algorithm, clustering is achieved by minimizing the sum of all pair-wise (euclidean)distances between observations in each cluster such that all the elements within a cluster are as similar as possible. There are three major steps in kmeans clustering. The first is the initial step where each observation is assigned to one of the k clusters. Next, is the calculation of the distances of each observation to the k random cluster centers. All the points closest to each cluster center forms a cluster. The process is repeated until the cluster assignments stop changing.

Principal Component Analysis

I also performed PCA on the data to find a low dimensional representation of the observations that explains a good amount of variance in the data. The outcome of this analysis is the principal components (PC's) which are the axes produced by a linear combination of the original variables.

In addition to multivariate normality, the PCA algorithm requires that the data are scaled and that there are no outliers in the data. These were all carried out as well.

PCA helped to reduce the dimensionality and creating new features (PC's) that explained significant amount of variance in the data.

```
# TITLE: Unsupervised Machine Learning on Rare Earth Element Data set

library(vegan) # for clustering
```

```
## Loading required package: permute
```

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

```
## This is vegan 2.6-2
```

```
library(cluster) # for clustering
library(shinythemes)
library(shinyWidgets)
library(shiny)
library(shinydashboard)
```

```
##
## Attaching package: 'shinydashboard'
```

```
## The following object is masked from 'package:graphics':
##
##     box
```

```
library(recipes)
```

```
##
## Attaching package: 'recipes'
```

```
## The following object is masked from 'package:permute':
##
##     check
```

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

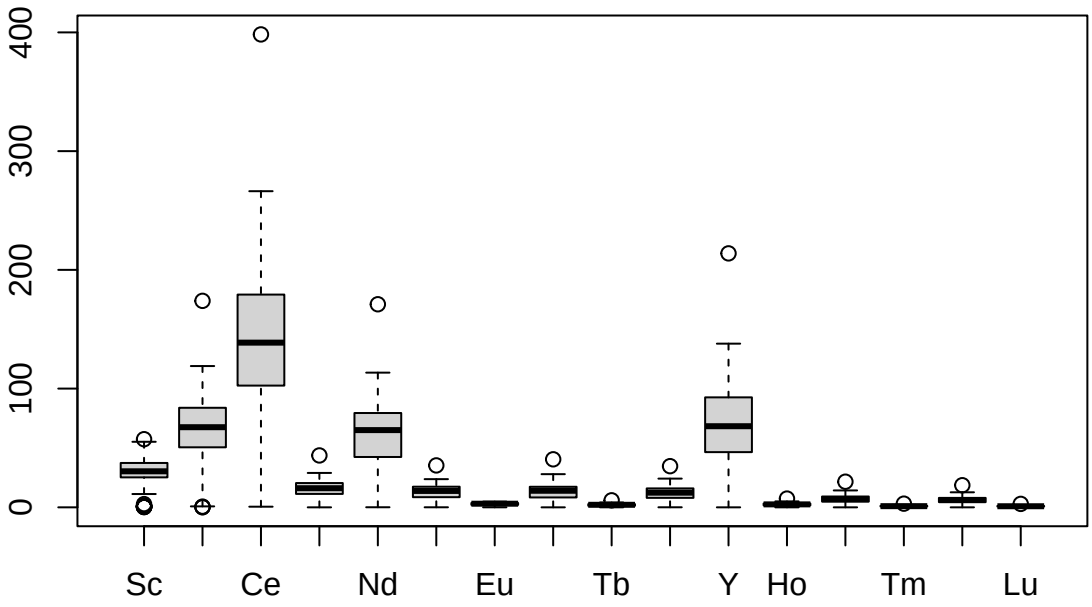
```
library(readr)


# PART ONE : Principal Component Analysis (PCA)
# Data Preparation

rareEarthMetals <-read.table('mlData2.csv', header=TRUE, sep=',', row.names=1, skip=1) #extracted the f
extract <- rareEarthMetals[, 6:21]
extract[is.na(extract)] <- 0 #replace NAs with zero otherwise the PCA will not work
colnames(extract) <- c('Sc', 'La', 'Ce', 'Pr', 'Nd', 'Sm', 'Eu', 'Gd', 'Tb', 'Dy', 'Y', 'Ho', 'Er', 'Tm
```

I checked for outliers to avoid having a skewed data distribution. The PCA algorithm is also very sensitive to outliers, hence it was important to remove the outliers

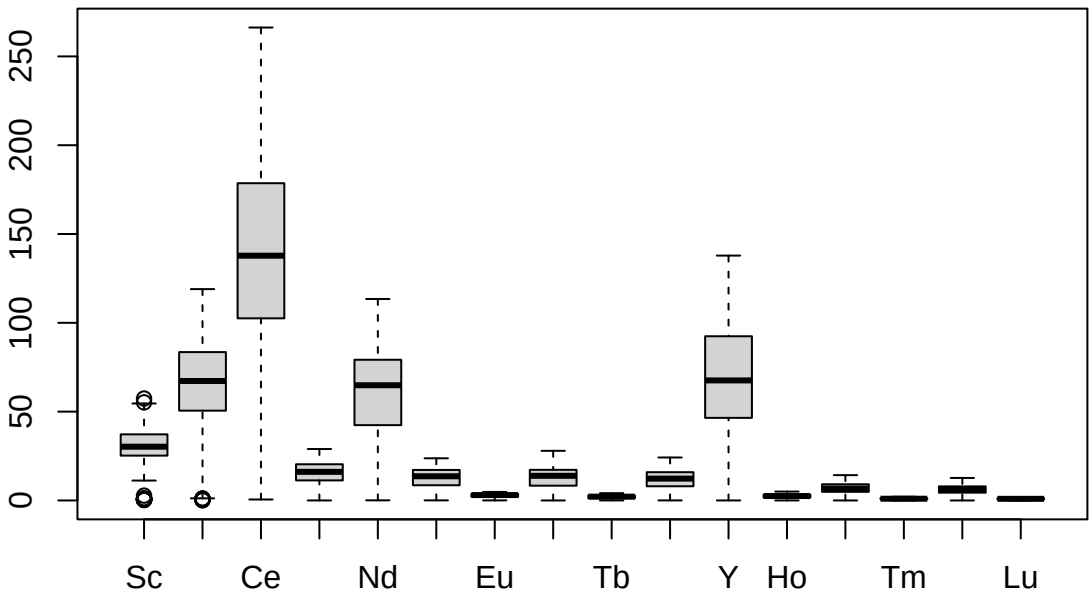
```
# Check for outliers
#dev.new()
boxplot(extract)
```



When I discovered there were outliers, I worked on eliminating them and rechecked that there were no outliers.

```
# Eliminate outliers
Q_Ce <- quantile(extract$Ce, probs=c(.25, .75), na.rm = FALSE)
iqr <- IQR(extract$Ce)
up <-  Q_Ce[2]+1.5*iqr # Upper Range
low<- Q_Ce[1]-1.5*iqr # Lower Range
ree_NoOutliers<- subset(extract, extract$Ce > low & extract$Ce < up)

# Re-check for outliers
#dev.new()
boxplot(ree_NoOutliers)
```

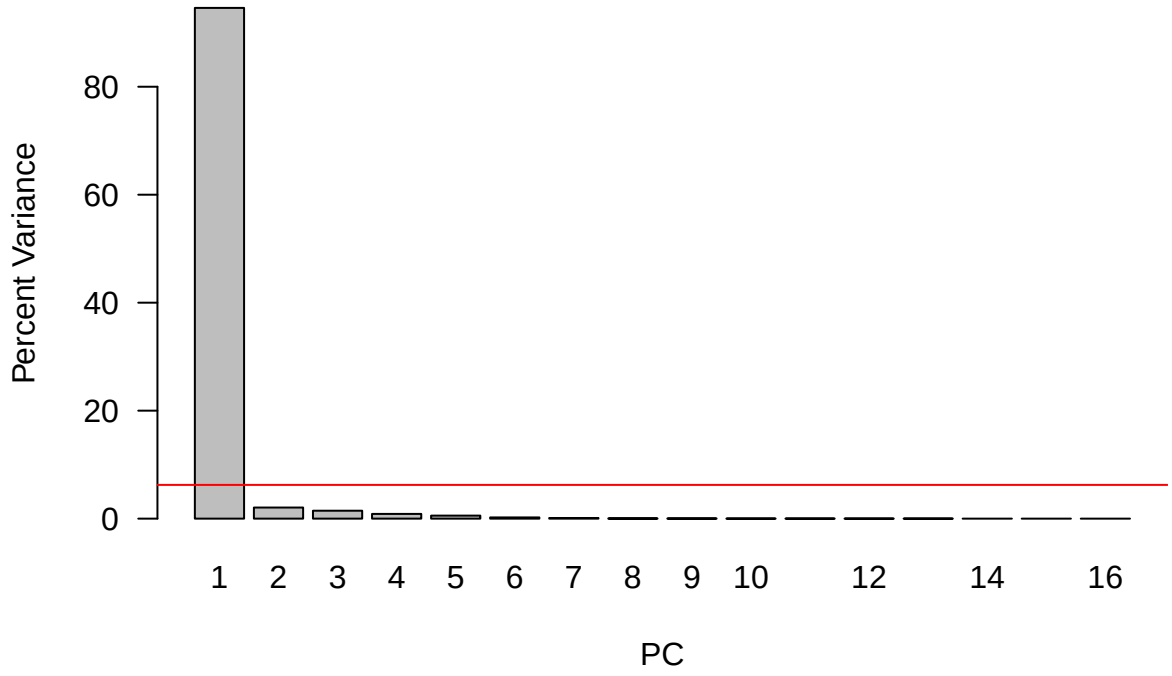


I ran the PCA of the dataset without outliers then checked the percent variance of the PCs and plotted the result on the scree plot as shown below.

```
# RUN THE PCA
reePCA <- prcomp(ree_NoOutliers, scale.=TRUE)
names(reePCA)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

```
# RENAME THE OUTPUT OF PCA TO MORE COMMON NAMES
sd <- reePCA$sdev
loadings <- reePCA$rotation
rownames(loadings) <- colnames(ree_NoOutliers)
scores <- reePCA$x
# SHOW SCREEPLOT OF VARIATION
var <- sd^2
varPercent <- var/sum(var) * 100
#dev.new()
barplot(varPercent, xlab='PC', ylab='Percent Variance', names.arg=1:length(varPercent), las=1, ylim=c(0
abline(h=1/ncol(ree_NoOutliers)*100, col='red') # show PCs that explain one variable worth of variance
```




```
# SHOW LOADINGS
loadings
```

```
##           PC1          PC2         PC3         PC4          PC5         PC6
## Sc -0.2410589  0.056601224 -0.55726435 -0.11686901  0.658805297  0.40021168
## La -0.2466114  0.289363422  0.36828856 -0.17382125  0.353684587 -0.27232247
## Ce -0.2498740  0.203513230  0.36249156 -0.19610320  0.170276729 -0.10969390
## Pr -0.2505137 -0.010504877  0.36050348  0.30146027  0.244563222 -0.02472314
## Nd -0.2532181  0.213435364  0.18022302 -0.11623875 -0.091550383  0.29695996
## Sm -0.2538379  0.161349279  0.04171747 -0.09633171 -0.244167230  0.40344277
## Eu -0.2288631  0.633095044 -0.39158121  0.48217672 -0.165863209 -0.33669703
## Gd -0.2511341 -0.080054620  0.20299319  0.36087258 -0.163591525  0.48127981
## Tb -0.2546033  0.044376400 -0.06705917 -0.17585414 -0.351546521  0.09246762
## Dy -0.2541301 -0.009415186 -0.13083554 -0.25727536 -0.292490217 -0.06295651
```

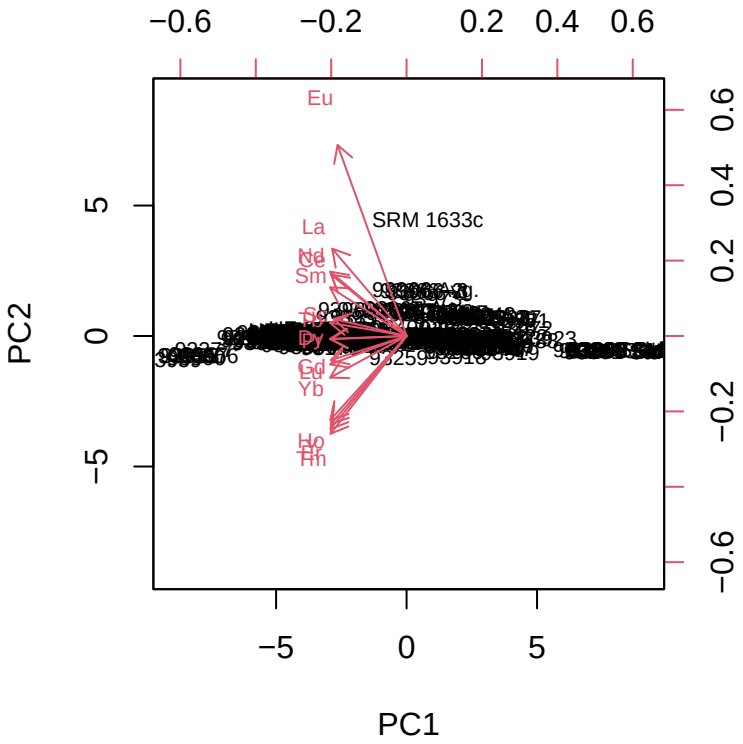
```
## Y  -0.2512463 -0.292025964 -0.07703284  0.21086335  0.006585662 -0.10922202
## Ho -0.2528583 -0.278388380 -0.02502633  0.19941472 -0.021447875 -0.07951533
## Er -0.2522389 -0.308177924 -0.04197325  0.16379078  0.025926703 -0.16846695
## Tm -0.2518899 -0.324098406 -0.01102126  0.11938086  0.071953930 -0.13519364
## Yb -0.2535117 -0.137933301 -0.12218636 -0.30783745 -0.077002550 -0.21914885
## Lu -0.2531233 -0.094099062 -0.13677118 -0.35226469 -0.091584605 -0.17784850
##              PC7         PC8         PC9        PC10        PC11        PC12
## Sc  0.0008444264 -0.09426327  0.08197375 -0.01702359  0.02306842 -0.04013196
## La -0.1685518465 -0.27617827 -0.40339466 -0.12808578  0.32988946  0.12397355
## Ce  0.0961459077 -0.20143260  0.46820939  0.23887589 -0.59174755  0.06173868
## Pr -0.0170229898  0.37651462  0.21119839 -0.11705600  0.26914645 -0.32393334
## Nd  0.0224894889  0.33087593 -0.17963876  0.06222527  0.02531462 -0.08110713
## Sm -0.3313645795  0.40649386  0.07866842  0.00786423 -0.03991734  0.33600731
## Eu  0.1065758935  0.04614709 -0.01176796  0.01132749 -0.06989978  0.03635769
## Gd  0.4144850438 -0.40097860 -0.29193307 -0.06743727 -0.13012405 -0.05073934
## Tb -0.1097358726 -0.33782698  0.19301381 -0.15325393  0.26084370 -0.23028306
## Dy -0.1527426052 -0.27348735  0.18367942  0.10263007  0.26409829  0.14708211
## Y  -0.6246514869 -0.07460135 -0.38558603  0.15708752 -0.39405748 -0.08449066
## Ho -0.0698778750 -0.07372804  0.27093837 -0.04886688  0.06874732 -0.16940360
## Er  0.0206206661  0.07755558  0.21719203  0.05337184  0.12138725 -0.08114760
## Tm  0.2734694102  0.08336462 -0.02293054  0.10528731  0.14848226  0.72248633
## Yb  0.1943352734  0.18879648 -0.11815245 -0.73817508 -0.32870785 -0.02913597
## Lu  0.3537931862  0.21526706 -0.29693639  0.53292582  0.03684854 -0.33313638
##             PC13         PC14          PC15         PC16
## Sc  0.034258487 -0.013249861  0.0222332136 -0.020227915
## La -0.239391501 -0.001466086 -0.1490507181  0.052996102
## Ce  0.053240851 -0.042513620 -0.0060146427  0.022494320
## Pr  0.169004017 -0.202961063  0.3722909279 -0.264159924
## Nd  0.435016875  0.548510207 -0.1690921130  0.275184071
## Sm -0.449438631 -0.262338641 -0.0733434646 -0.064407926
## Eu -0.003152396  0.001503939  0.0002082286  0.011477609
## Gd -0.127188662 -0.024613446 -0.0029569564 -0.205042795
## Tb  0.295838312 -0.461318088 -0.0415401122  0.394475533
## Dy  0.089216540  0.377896455  0.2550305629 -0.559395178
## Y   0.207678504 -0.065860389  0.0998862196 -0.015900131
## Ho -0.507880457  0.421878979  0.1915482519  0.468224380
## Er  0.012482348  0.004002951 -0.8079037771 -0.230430601
## Tm  0.248999047 -0.121604549  0.1600991387  0.232129199
## Yb -0.011349636  0.022296767  0.0426703632 -0.097159594
## Lu -0.209431360 -0.182450511  0.1052496912 -0.001959104
```

```
sqrt(1/ncol(ree_NoOutliers)) #cut-off for important loadings
```

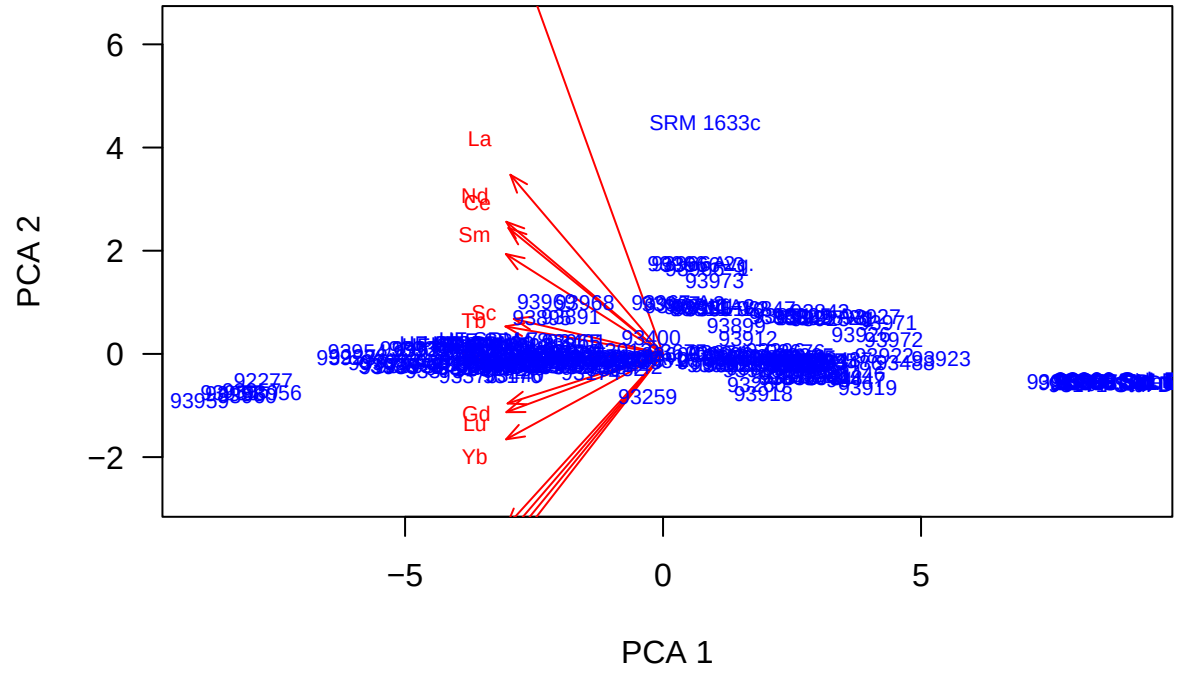
```
## [1] 0.25
```

Afterwards, I did a biplot for the important PC that shows the influence of an increase in a given variable

```
# Biplot for important PCs ; PC1 & PC2
#dev.new()
biplot(scores[, 1:2], loadings[, 1:2], cex=0.7)
```

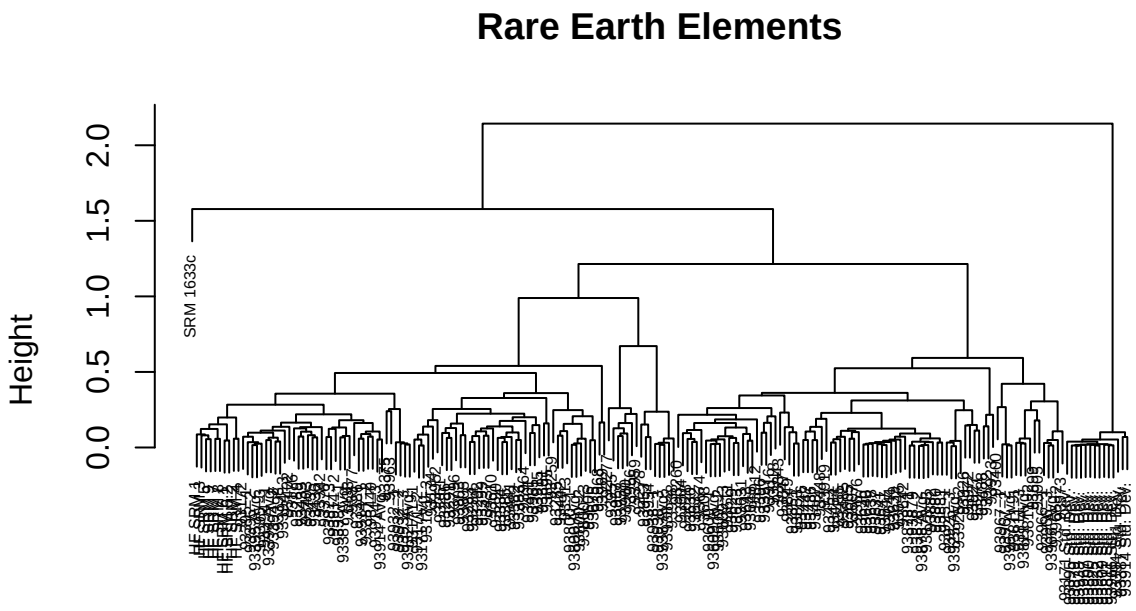



```
#or DIY Biplot
#dev.new(height=7, width=7)
plot(scores[, 1], scores[, 2], xlab='PCA 1', ylab='PCA 2', type='n', asp=1, las=1)#asp sets aspect rati
scaling <- 12
textNudge <- 1.2
arrows(0, 0, loadings[, 1]* scaling, loadings[, 2]* scaling, length=0.1, angle=20, col='red')
text(loadings[, 1]*scaling*textNudge, loadings[, 2]*scaling*textNudge, rownames(loadings), col='red', c
text(scores[, 1], scores[, 2], rownames(scores), col='blue', cex=0.7)
```



I applied kmeans algorithm on my data set. Also applied hierachical clustering and achieved similar results.

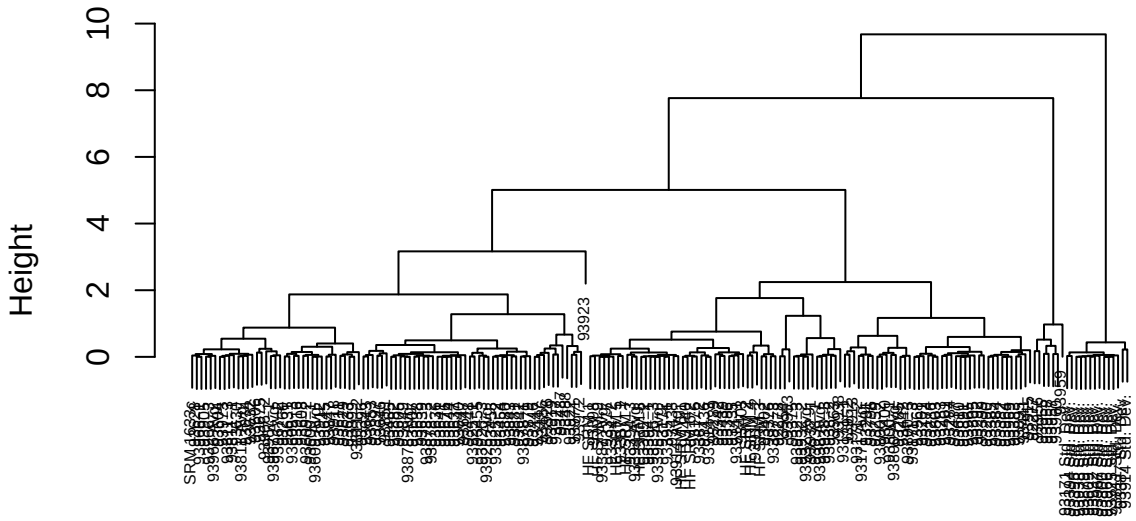
```
#PART TWO : CLUSTERING - original samples without outliers
reeT1 <- decostand(ree_NoOutliers, method='max') # transformation to make values go from zero to one
reeT1Agnes <- agnes(reeT1)
# Plot the cluster
#dev.new()
plot(reeT1Agnes, which.plots = 2, main='Rare Earth Elements', cex=0.5)
```




```
#PART TWO (B1) : CLUSTERING USING PRINCIPAL COMPONENTS - HIERACHICHAL CLUSTERING
x_new <- cbind(scores[, 1])

#Just doing this for the Shiny App
PC1 <- scores[,1]
PC2 <- scores[,2]
PC3 <- scores[,3]

reeT1_withPCA <- decostand(x_new, method='max') # transformation to make values go from zero to one
reeT1Agnes_withPCA <- agnes(x_new)
# Plot the cluster
#dev.new()
plot(reeT1Agnes_withPCA, which.plots = 2, main='Rare Earth Elements', cex=0.5)
```




```
# Let's get the 3 Clusters out
cluster1 <- reeT1Agnes_withPCA$order[which(reeT1Agnes_withPCA$order.lab=='SRM 1633c') : which(reeT1Agnes
cluster2 <- reeT1Agnes_withPCA$order[which(reeT1Agnes_withPCA$order.lab=='HF SRM 1') : which(reeT1Agnes_
cluster3 <- reeT1Agnes_withPCA$order[which(reeT1Agnes_withPCA$order.lab=='93171 Std. Dev.') : which(reeT
# Get the size of the three apparent clusters
ree_cluster1 <- ree_NoOutliers[cluster1, ]
ree_cluster2 <- ree_NoOutliers[cluster2, ]
ree_cluster3 <- ree_NoOutliers[cluster3, ]
length(rownames(ree_cluster1))
```

```
## [1] 85
```

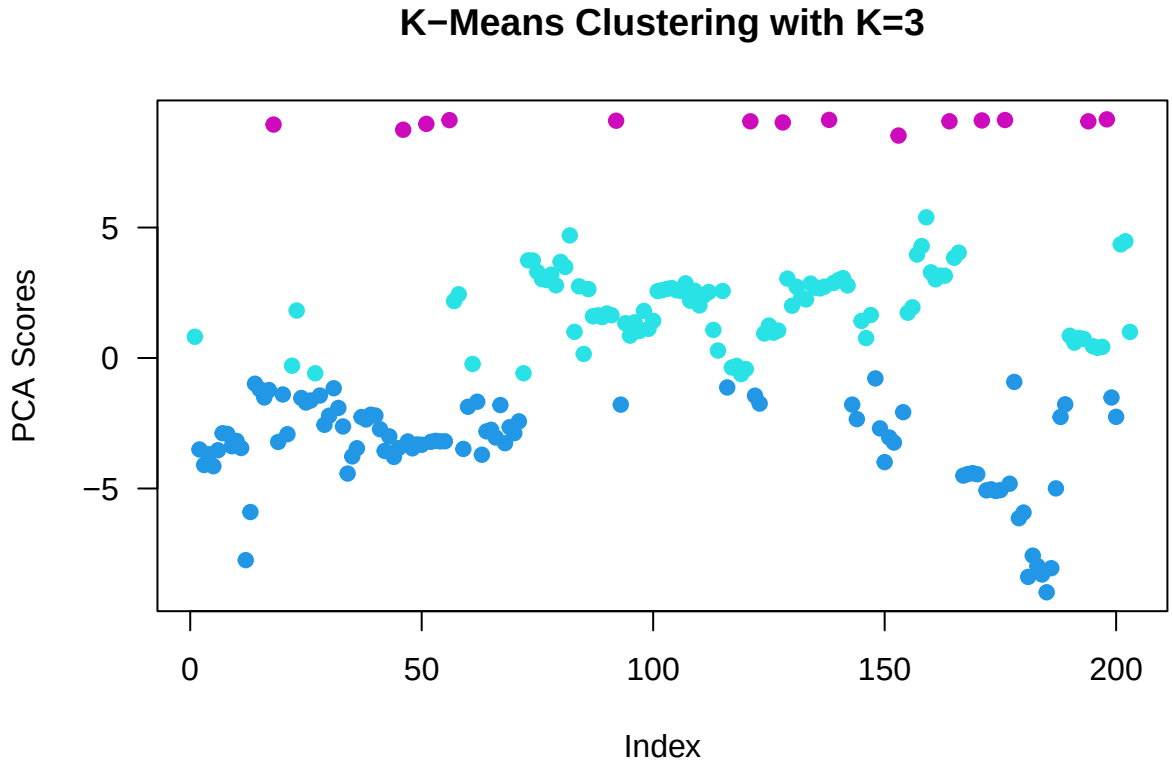
```
length(rownames(ree_cluster2))
```

```
## [1] 103
```

```
length(rownames(ree_cluster3))
```

```
## [1] 14
```

```
#PART TWO (B2) : CLUSTERING USING PRINCIPAL COMPONENTS - K-Means CLUSTERING
ree_kmeans <- kmeans(x_new, 3, nstart=20) #suggesting 3 clusters here because the clustering from the o
# Plot the cluster
#dev.new()
plot(x_new, col=(ree_kmeans$cluster+3), las=1, ylab='PCA Scores', main='K-Means Clustering with K=3', p
```




```
# Get the size of the three clusters
ree_kmeans$size[1]
```

```
## [1] 95
```

```
ree_kmeans$size[2]
```

```
## [1] 94
```

```
ree_kmeans$size[3]
```

```
## [1] 14
```

```
REE_PC<-cbind(ree_NoOutliers, PC1, PC2, PC3)
head(REE_PC)
```

```
##              Sc    La     Ce    Pr    Nd    Sm   Eu    Gd   Tb    Dy      Y
## SRM 1633c 37.60 87.00 180.00  0.00 87.00 19.00 4.67  0.00 3.12 18.70   0.00
## HF SRM 1  38.10 78.86 177.31 20.49 86.06 19.33 4.21 20.30 2.97 16.58  98.61
## HF SRM 2  37.73 84.35 188.06 21.91 86.99 19.39 4.51 20.52 3.12 17.95 103.03
## HF SRM 3  37.82 82.12 179.83 21.11 83.63 18.81 4.44 19.92 3.03 17.23 100.85
## HF SRM 4  35.40 84.96 191.04 22.13 87.41 20.04 4.54 20.84 3.18 17.95 104.62
## HF SRM 5  34.83 78.51 179.92 21.21 84.29 18.88 4.34 20.04 3.06 17.34  94.61
##              Ho   Er   Tm   Yb   Lu        PC1       PC2        PC3
## SRM 1633c 0.00 0.00 0.00 7.70 1.32  0.8140255 4.4875556 -1.6866268
## HF SRM 1  3.38 9.27 1.28 7.82 1.17 -3.5040725 0.1484885 -0.2920248
## HF SRM 2  3.56 9.87 1.37 8.29 1.23 -4.1011287 0.1636880 -0.2593745
## HF SRM 3  3.41 9.53 1.31 8.00 1.19 -3.6894571 0.1981460 -0.3440235
## HF SRM 4  3.55 9.90 1.35 8.39 1.21 -4.1456856 0.2054144 -0.1045645
## HF SRM 5  3.44 9.41 1.29 7.95 1.19 -3.5341819 0.1770619 -0.1829811
```

4 Creating a Shiny Application

Here, I installed the packages for shiny application, shiny widgets and DT

The shiny app will show the results of the clusters of these elements especially when comparing the relationship between two of these elements at a time. These variables can be varied.

For example, looking at Nd and Dy, the blue cluster corresponds to high Nd, high Dy. But this same blue cluster corresponds to low Ho and low Sm. With this, we can extract very useful information about our data and what each cluster or sample represents.

```
library(shinythemes)
library(shinyWidgets)
library(shiny)
library(shinydashboard)
library(recipes)
library(dplyr)
library(tidyr)
library(readxl)

library(readxl)
library(vegan) # for clustering
library(cluster) # for clustering
library(readr)



# Define UI for application
ui = fluidPage(
    navbarPage("Chinenye Chinwego's Clustering App",
               tabPanel("About",
                        tabName = "welcome",
                        icon=icon("home"),

                        fluidPage(theme=shinytheme("united"),
                                  h1("Welcome!"),
                                  br(),
                                  br(),
                                  br(),
                                  br(),
                                  p(strong("This App is about:")),

                                  h1("Rare Earth Element Composition Analysis"),
                                  br(),
                                  p(strong("The Algorithim used for this analysis is CLUSTERING")),
                                  p(tags$cite("Clustering is unsupervised Machine learning, thus the infor
                                  p(tags$cite("I performed dimensionality reduction on the features by co
                                  p(tags$cite("This app will let us apply clustering on either the origina
                                  p(tags$cite("which will help us to understand the data better and create

                                  br(),
                                  br(),
                                  br(),
                                  br(),
                                  br(),
```

```
                                p("The data set was obtained from a paper titled:"),

                                p(tags$blockquote("Trends in the Rare Earth Element Content of U.S.-Base
                                p(tags$cite("by")),
                                p(tags$body("Ross K. Taggart, James C. Hower, Gary S. Dwyer, and Heileen
                                p(tags$body("Environmental Science & Technology 2016 50 (11), 5919-5926
                                 p(tags$body("DOI: 10.1021/acs.est.6b00085")),



                      )),



          tabPanel("Cluster",
                   icon=icon("chart-bar"),


# sidebarLayout(
  ui <- pageWithSidebar(
  headerPanel('clustering (k-means)'),
  sidebarPanel(
    selectInput('xcol', 'X Variable', names(REE_PC)),
    selectInput('ycol', 'Y Variable', names(REE_PC),
                selected=names(REE_PC)[[2]]),
    numericInput('clusters', 'Cluster count', 3,
                 min = 5, max = 5)
  ),
  mainPanel(
    plotOutput("plot1"),


             )
  ))))
 server <- function(input, output, session) {
# Define server logic
 #server <- function(input, output) {
    # Combine the selected variables into a new data frame
   selectedData <- reactive({
     REE_PC[, c(input$xcol, input$ycol)]
   })
   clusters <- reactive({
     kmeans(selectedData(), input$clusters)
   })
   output$plot1 <- renderPlot({
     palette(c("#999999", "#E69F00", "#56B4E9", "#009E73",
               "#F0E442", "#0072B2", "#D55E00", "#CC79A7"))
     par(mar = c(5.1, 4.1, 0, 1))
     plot(selectedData(),
          col = clusters()$cluster,
          pch = 20, cex = 3)
     points(clusters()$centers, pch = 16, cex = 2, lwd = 4)
   })
```

```
 }



# Run the application
shinyApp(ui = ui, server = server)
```

5 Deploying your Application

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

```
##
## Attaching package: 'rsconnect'
```

```
## The following object is masked from 'package:shiny':
##
##     serverInfo
```

```
rsconnect::setAccountInfo(name='sd6b6c-chinenye-chinwego', token='94437EA32928AA288E020CEFA8185A32', se
```