Principal Component Analysis

Vaishnavi

2025-02-02

Table of Contents

library(haven)  
library(corrplot)

## corrplot 0.95 loaded

library(PerformanceAnalytics)

## Loading required package: xts

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

##   
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':  
##   
## legend

library(FactoMineR)  
library(dplyr)

##   
## ######################### Warning from 'xts' package ##########################  
## # #  
## # The dplyr lag() function breaks how base R's lag() function is supposed to #  
## # work, which breaks lag(my\_xts). Calls to lag(my\_xts) that you type or #  
## # source() into this session won't work correctly. #  
## # #  
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #  
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop #  
## # dplyr from breaking base R's lag() function. #  
## # #  
## # Code in packages is not affected. It's protected by R's namespace mechanism #  
## # Set `options(xts.warn\_dplyr\_breaks\_lag = FALSE)` to suppress this warning. #  
## # #  
## ###############################################################################

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:xts':  
##   
## first, last

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(weights)

## Loading required package: Hmisc

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':  
##   
## src, summarize

## The following objects are masked from 'package:base':  
##   
## format.pval, units

library(ggplot2)  
library(hrbrthemes)  
library(srvyr)

##   
## Attaching package: 'srvyr'

## The following object is masked from 'package:Hmisc':  
##   
## summarize

## The following object is masked from 'package:stats':  
##   
## filter

library(DescTools)

## Registered S3 method overwritten by 'DescTools':  
## method from   
## reorder.factor gdata

##   
## Attaching package: 'DescTools'

## The following objects are masked from 'package:Hmisc':  
##   
## %nin%, Label, Mean, Quantile

library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

## The following object is masked from 'package:DescTools':  
##   
## Recode

## The following object is masked from 'package:dplyr':  
##   
## recode

#### The dataste beging used is the All India Debt and Investment Survey, 2019. The goal is to run a PCA on variables relating to investment and expenditure in rural India.

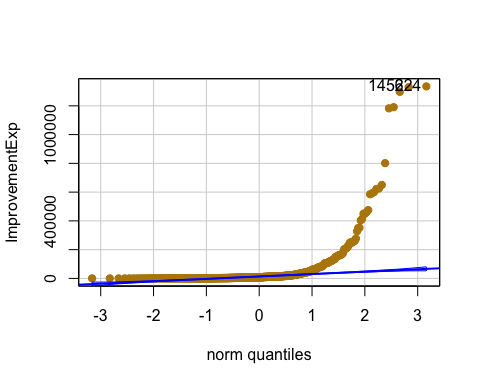
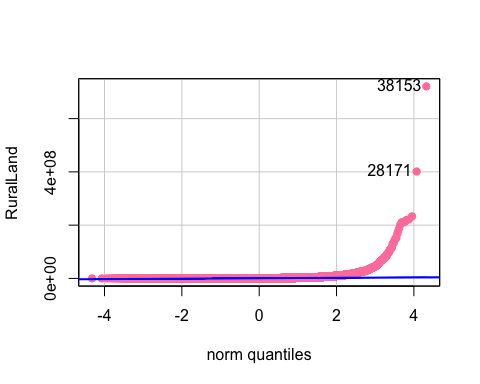
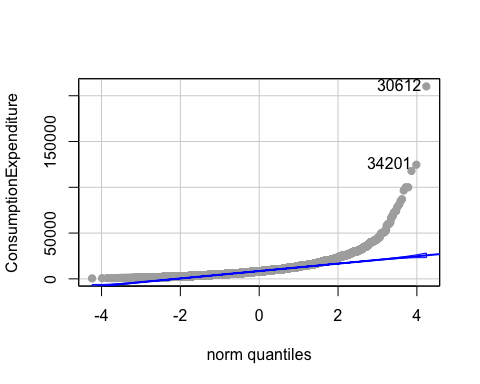
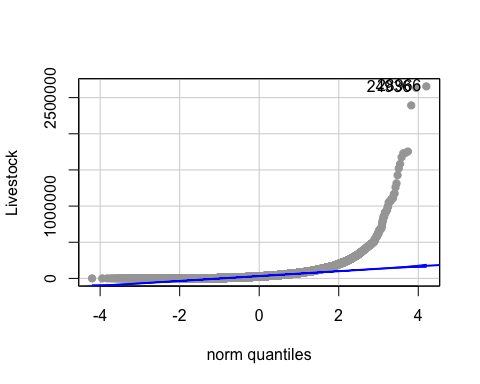
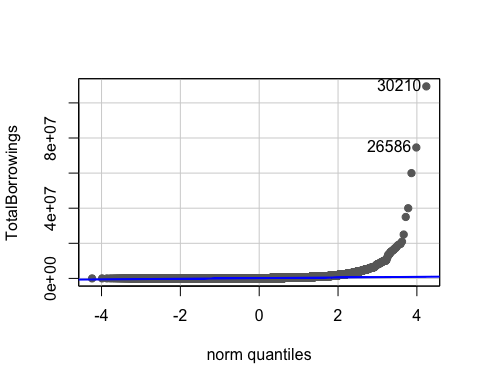
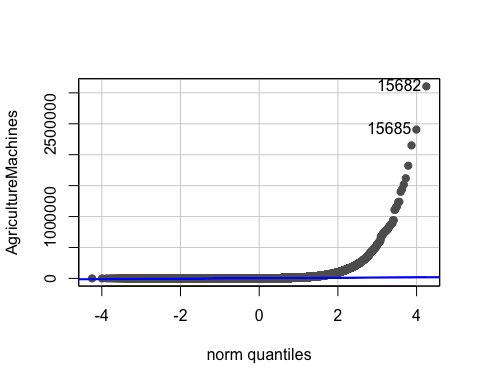
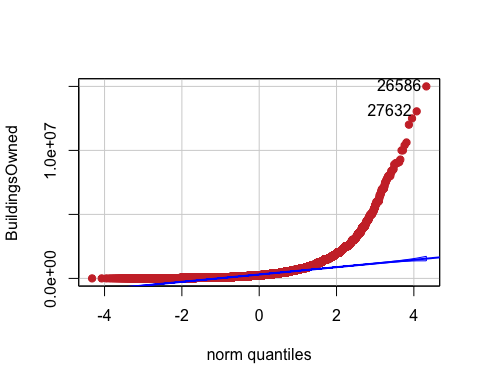
####Part 1- Loading the dataset and cleaning

AIDIS <- read\_dta("~/Desktop/Yale/Multivariate Statistics/AIDIS\_for\_multistats.dta")  
AIDIS <- AIDIS[AIDIS$Sector == "1",] #keep rural only  
AIDIS <- AIDIS %>%  
 mutate(ImprovementExp = FarmImprovementExp + NonFarmImprovement)  
#creating a new dataset with relevant variables only  
fv<- AIDIS[ ,c("ImprovementExp", "BuildingsOwned", "usual\_monthly\_con\_exp", "livestock\_owned\_value", "borrowings" , "agri\_machine\_owned\_value", "rural\_land\_value")]  
# Rename multiple variables at once  
fv <- fv %>%  
 rename(  
 ConsumptionExpenditure = usual\_monthly\_con\_exp,  
 Livestock = livestock\_owned\_value,  
 TotalBorrowings = borrowings,  
 AgricultureMachines = agri\_machine\_owned\_value,  
 RuralLand = rural\_land\_value  
 )  
  
#Checking for missing values  
missing\_values <- sapply(fv, function(x) sum(is.na(x)))  
print(missing\_values)

## ImprovementExp BuildingsOwned ConsumptionExpenditure   
## 68588 4504 24618   
## Livestock TotalBorrowings AgricultureMachines   
## 31358 24618 22996   
## RuralLand   
## 4848

####Part 2- Normality and linear relaionships

#Pre-Transformation  
# List of variables  
variables <- c("BuildingsOwned", "AgricultureMachines", "TotalBorrowings",   
 "Livestock", "ConsumptionExpenditure", "RuralLand", "ImprovementExp")  
  
# Loop through each variable and create a QQ plot  
for (var in variables) {  
 qqPlot(fv[[var]],   
 col = sample(colors(), 1), # Random color for each plot  
 pch = 19,   
 ylab = var) # Automatically set y-axis label as the column name  
   
   
}

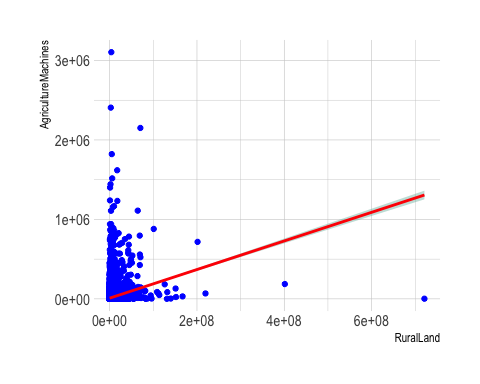


## Correlation between Rural land value and value of agriculture machine owned  
ggplot(fv, aes(x = RuralLand, y = AgricultureMachines)) +  
 geom\_point(color = 'blue') + # Plot points with blue color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.95) + # Add linear trend with CI  
 theme\_ipsum()

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: Removed 24978 rows containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 24978 rows containing missing values or values outside the scale range  
## (`geom\_point()`).

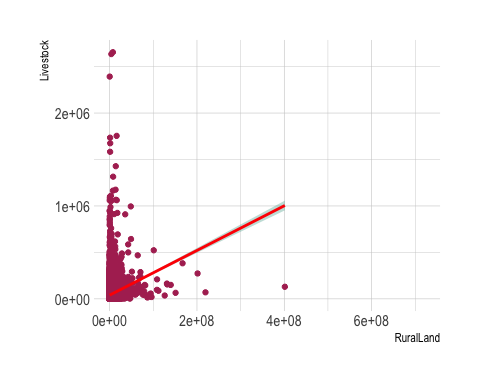


## Correlation between Rural land value and value of livestock owned  
ggplot(fv, aes(x = RuralLand, y = Livestock)) +  
 geom\_point(color = 'maroon') + # Plot points with maroon color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.95) + # Add linear trend with CI  
 theme\_ipsum()

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: Removed 32910 rows containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 32910 rows containing missing values or values outside the scale range  
## (`geom\_point()`).

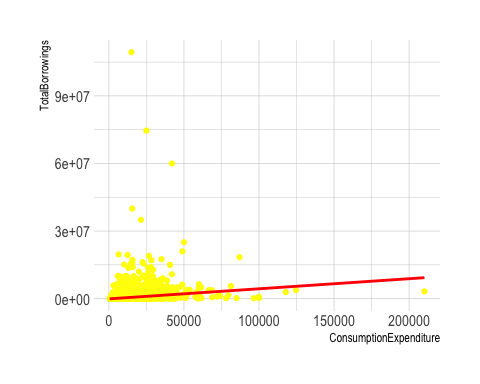


## Correlation between Monthly consumption expenditrure and borrowings  
ggplot(fv, aes(x = ConsumptionExpenditure, y = TotalBorrowings)) +  
 geom\_point(color = 'yellow') + # Plot points with yello color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.99) + # Add linear trend with CI  
 theme\_ipsum()

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: Removed 24618 rows containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 24618 rows containing missing values or values outside the scale range  
## (`geom\_point()`).

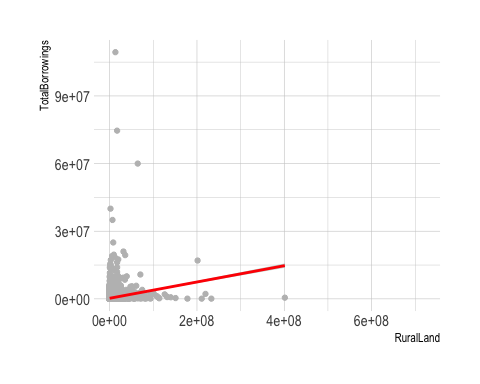


## Correlation between Rural land value and borrowings  
ggplot(fv, aes(x = RuralLand, y = TotalBorrowings)) +  
 geom\_point(color = 'grey') + # Plot points with grey color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.99) + # Add linear trend with CI  
 theme\_ipsum()

## `geom\_smooth()` using formula = 'y ~ x'

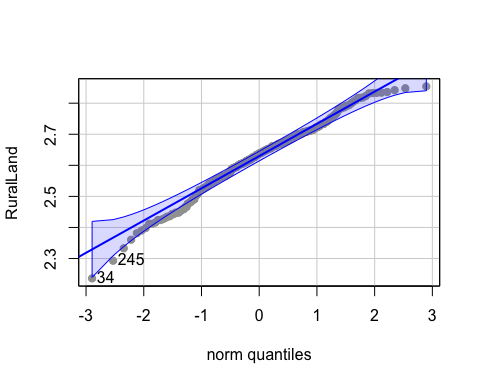
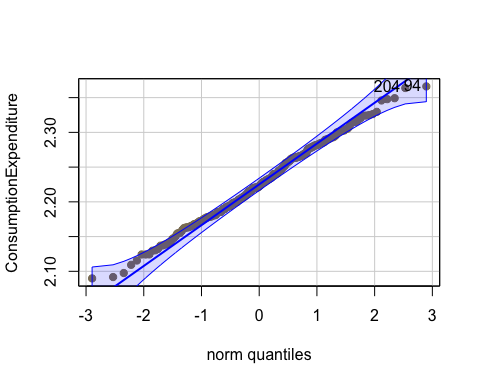
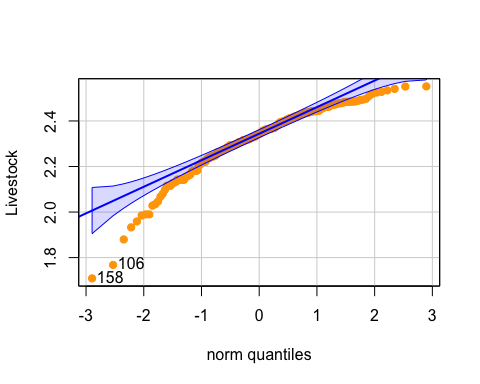
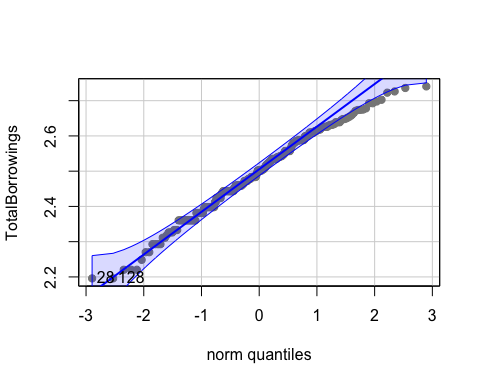
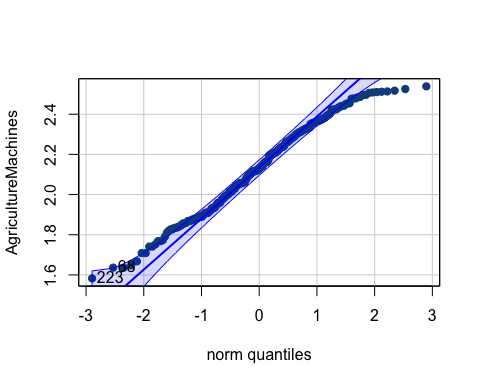
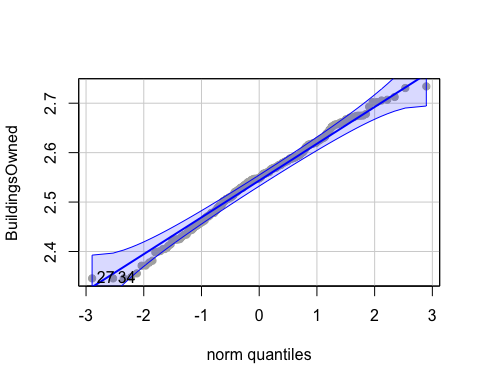
## Warning: Removed 27322 rows containing non-finite outside the scale range  
## (`stat\_smooth()`).

## Warning: Removed 27322 rows containing missing values or values outside the scale range  
## (`geom\_point()`).



####Part 3- Transformation ans normality

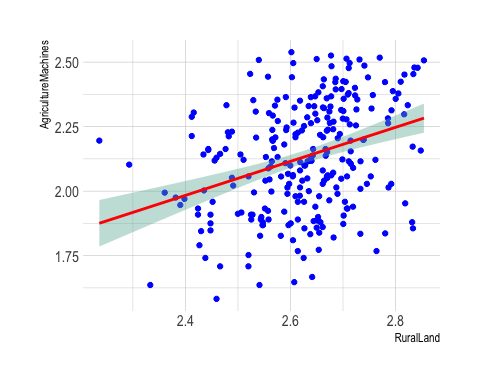
##Tranformations are needed.  
fv[] <- lapply(fv, function(x) as.numeric(as.character(x)))# Remove any rows with NAs (if necessary)  
fv\_clean <- na.omit(fv)  
  
fv\_log <- fv\_clean # Copying the data frame to preserve the original  
fv\_log[] <- lapply(fv\_log, function(x) log(x)) #log transformation  
# If there are any zero or negative values replacing them with NA or a small value):  
fv\_log[] <- lapply(fv\_log, function(x) {  
 if (is.numeric(x)) log(pmax(x, 1e-6)) else x  
})  
# Winsorize the Buildings owned as there are many outliers  
fv\_log$BuildingsOwned <- Winsorize(fv\_log$BuildingsOwned, val= Quantile(fv\_log$BuildingsOwned, probs = c(0.01, 1))) # Cap values below 1st percentile and above 99th percentile  
  
##Univariate Normality for transofrmed data  
  
# List of variables  
variables <- c("BuildingsOwned", "AgricultureMachines", "TotalBorrowings", "Livestock", "ConsumptionExpenditure", "RuralLand")  
  
# Loop through each variable and create a QQ plots for all  
for (var in variables) {  
 qqPlot(fv\_log[[var]],   
 col = sample(colors(), 1), # Random color for each plot  
 pch = 19,   
 ylab = var) # Automatically set y-axis label as the column name  
}



####Part 4- Establishing Correlation

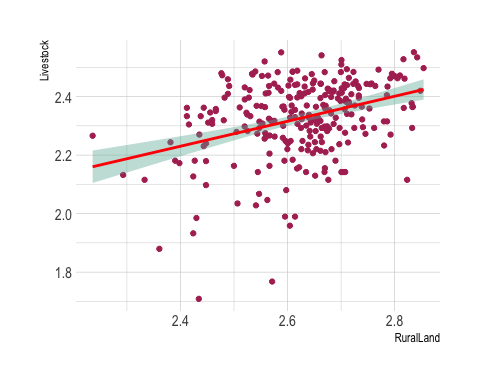
#Looking for correlation between the trasnformed variables  
  
## Correlation between Rural land value and value of agriculture machine owned  
ggplot(fv\_log, aes(x = RuralLand, y = AgricultureMachines)) +  
 geom\_point(color = 'blue') + # Plot points with blue color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.95) + # Add linear trend with CI  
 theme\_ipsum() # Apply theme

## `geom\_smooth()` using formula = 'y ~ x'



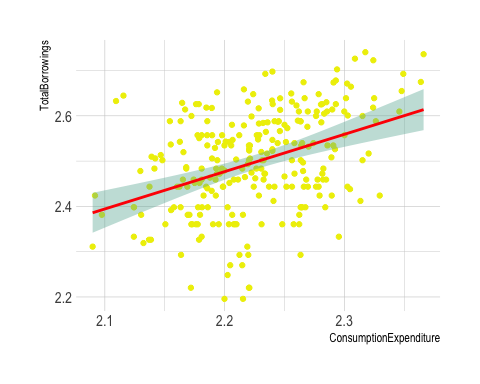
## Correlation between Rural land value and value of livestock owned  
ggplot(fv\_log, aes(x = RuralLand, y = Livestock)) +  
 geom\_point(color = 'maroon') + # Plot points with blue color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.95) + # Add linear trend with CI  
 theme\_ipsum() # Apply theme

## `geom\_smooth()` using formula = 'y ~ x'



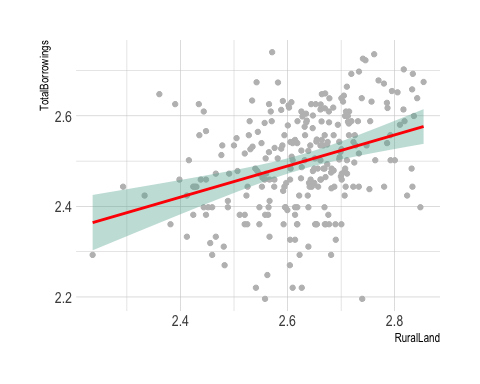
## Correlation between Monthly consumption expenditrure and borrowings  
ggplot(fv\_log, aes(x = ConsumptionExpenditure, y = TotalBorrowings)) +  
 geom\_point(color = 'yellow2') + # Plot points with blue color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.99) + # Add linear trend with CI  
 theme\_ipsum() # Apply theme

## `geom\_smooth()` using formula = 'y ~ x'

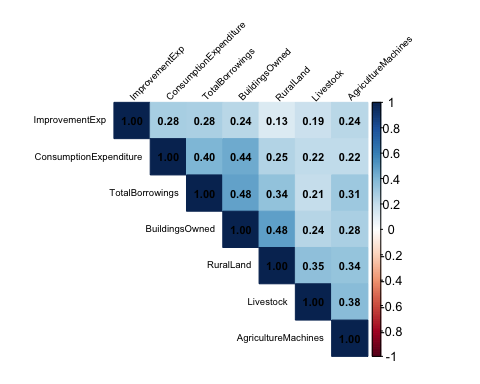


## Correlation between Rural land value and borrowings  
ggplot(fv\_log, aes(x = RuralLand, y = TotalBorrowings)) +  
 geom\_point(color = 'grey') + # Plot points with blue color  
 geom\_smooth(method = "lm", color = "red", fill = "#69b3a2", se = TRUE, level = 0.99) + # Add linear trend with CI  
 theme\_ipsum() # Apply theme

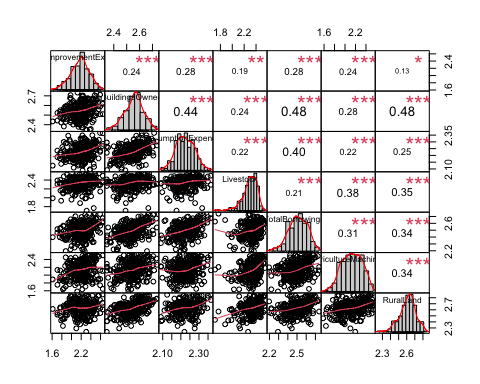
## `geom\_smooth()` using formula = 'y ~ x'



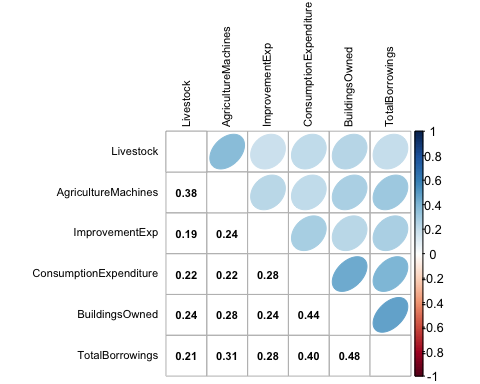
#building a correlation matrix  
cor\_matrix <- cor(fv\_log, use = "complete.obs")  
  
  
# Correlation plots with corrplot  
corrplot(cor\_matrix, method = "color", type = "upper", order = "hclust",   
tl.col = "black", tl.srt = 45, addCoef.col = "black", number.cex = 0.7, tl.cex = 0.6)



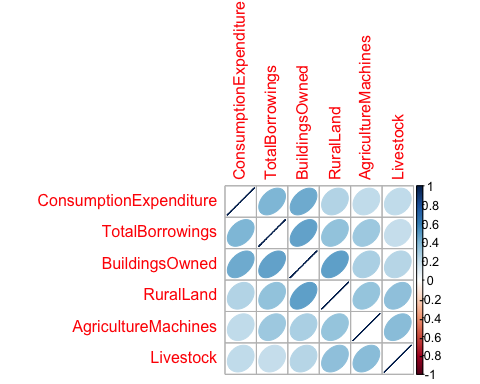
# Correlation charts  
chart.Correlation(fv\_log[,1:7])



# Correlation plots with corrplot  
corrplot.mixed(cor(fv\_log[,1:6]), lower.col = "black", upper = "ellipse", tl.col = "black", number.cex = .7, order = "hclust", tl.pos = "lt", tl.cex = .7)



corrplot(cor(fv\_log[,-1]),method = "ellipse", order="AOE")



####Part 5: PCA

pca1 <- prcomp(fv\_log[, 1:7], scale. = TRUE) #PCA and scaling the variables   
pca1

## Standard deviations (1, .., p=7):  
## [1] 1.6831653 1.0030975 0.9486441 0.8156968 0.7917428 0.7315961 0.6583113  
##   
## Rotation (n x k) = (7 x 7):  
## PC1 PC2 PC3 PC4 PC5  
## ImprovementExp 0.2892932 0.3227242 0.7398798 -0.07953973 -0.502659644  
## BuildingsOwned 0.4432482 0.1953662 -0.3673361 -0.01063590 -0.209372279  
## ConsumptionExpenditure 0.3803832 0.4238125 -0.0309139 0.54697150 0.425346749  
## Livestock 0.3320540 -0.5730709 0.2176204 0.56432418 -0.009233187  
## TotalBorrowings 0.4172802 0.2912514 -0.1256216 -0.37131364 0.233127301  
## AgricultureMachines 0.3636449 -0.4130766 0.2886561 -0.47798718 0.482374829  
## RuralLand 0.3981568 -0.3069107 -0.4125828 -0.09785270 -0.485239742  
## PC6 PC7  
## ImprovementExp -0.06110661 -0.04482235  
## BuildingsOwned -0.13436760 0.75393817  
## ConsumptionExpenditure -0.33845359 -0.28299725  
## Livestock 0.41949086 0.13946913  
## TotalBorrowings 0.70296000 -0.19721586  
## AgricultureMachines -0.37935374 0.09349477  
## RuralLand -0.22276869 -0.53140732

summary(pca1) #this gives us standard deviation

## Importance of components:  
## PC1 PC2 PC3 PC4 PC5 PC6 PC7  
## Standard deviation 1.6832 1.0031 0.9486 0.81570 0.79174 0.73160 0.65831  
## Proportion of Variance 0.4047 0.1437 0.1286 0.09505 0.08955 0.07646 0.06191  
## Cumulative Proportion 0.4047 0.5485 0.6770 0.77208 0.86163 0.93809 1.00000

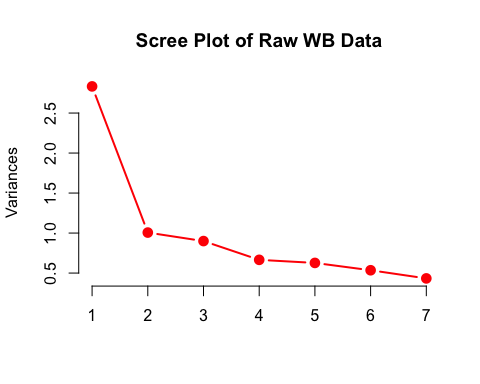
#Function that gives the variance   
summary.PCA.JDRS <- function(x){  
 sum\_JDRS <- summary(x)$importance  
 sum\_JDRS[1, ] <- sum\_JDRS[1, ]^2  
 attr(sum\_JDRS, "dimnames")[[1]][1] <- "Eigenvals (Variance)"  
 sum\_JDRS  
}   
  
sum(pca1$sdev^2) #get total variance

## [1] 7

############################Part 6: What components do we retain?##########################  
#How many components to retail?  
  
#1. Total variance explained by a given number of principle components  
#2. The ‘eigenvalue > 1’ criteria  
summary.PCA.JDRS(pca1)

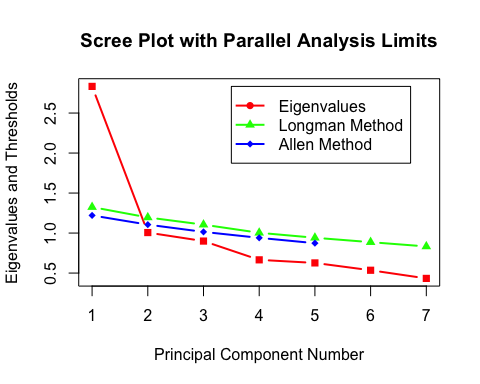
## PC1 PC2 PC3 PC4 PC5  
## Eigenvals (Variance) 2.833045 1.006205 0.8999255 0.6653613 0.6268567  
## Proportion of Variance 0.404720 0.143740 0.1285600 0.0950500 0.0895500  
## Cumulative Proportion 0.404720 0.548460 0.6770300 0.7720800 0.8616300  
## PC6 PC7  
## Eigenvals (Variance) 0.5352328 0.4333738  
## Proportion of Variance 0.0764600 0.0619100  
## Cumulative Proportion 0.9380900 1.0000000

#3.The ‘scree plot elbow’ method   
screeplot(pca1, type = "lines", col = "red", lwd = 2, pch = 19, cex = 1.2, main = "Scree Plot of Raw WB Data")



#function for a parallel plot  
source("http://reuningscherer.net/multivariate/r/parallel.R.txt")  
  
#4. Parallel Analysis: think about whether this is appropriate based on what you discover in question 1.  
parallelplot(pca1)

## pcompnum longman allen  
## 1 1 1.3247029 1.220634  
## 2 2 1.1953446 1.105421  
## 3 3 1.1050227 1.014720  
## 4 4 1.0035349 0.940353  
## 5 5 0.9413870 0.873631  
## 6 6 0.8863496 NA  
## 7 7 0.8326951 NA



#Get loadings  
round(pca1$rotation,2)

## PC1 PC2 PC3 PC4 PC5 PC6 PC7  
## ImprovementExp 0.29 0.32 0.74 -0.08 -0.50 -0.06 -0.04  
## BuildingsOwned 0.44 0.20 -0.37 -0.01 -0.21 -0.13 0.75  
## ConsumptionExpenditure 0.38 0.42 -0.03 0.55 0.43 -0.34 -0.28  
## Livestock 0.33 -0.57 0.22 0.56 -0.01 0.42 0.14  
## TotalBorrowings 0.42 0.29 -0.13 -0.37 0.23 0.70 -0.20  
## AgricultureMachines 0.36 -0.41 0.29 -0.48 0.48 -0.38 0.09  
## RuralLand 0.40 -0.31 -0.41 -0.10 -0.49 -0.22 -0.53

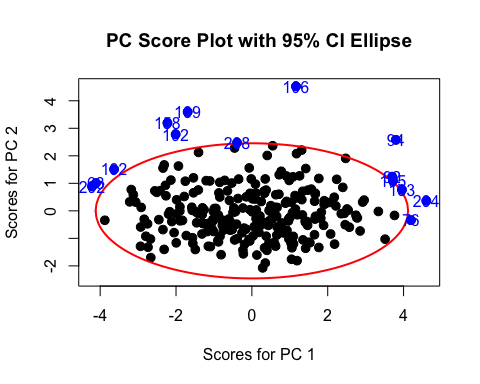
round(summary.PCA.JDRS(pca1), 3)

## PC1 PC2 PC3 PC4 PC5 PC6 PC7  
## Eigenvals (Variance) 2.833 1.006 0.900 0.665 0.627 0.535 0.433  
## Proportion of Variance 0.405 0.144 0.129 0.095 0.090 0.076 0.062  
## Cumulative Proportion 0.405 0.548 0.677 0.772 0.862 0.938 1.000

####Part 7: Score plots and Bi-plots

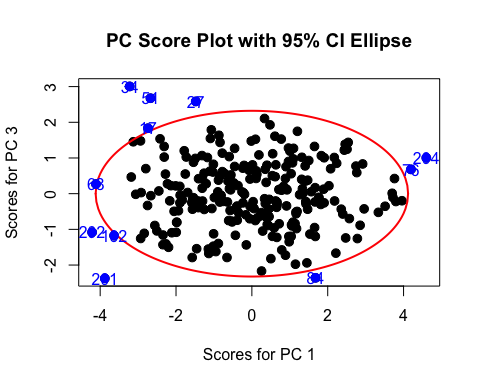
#function for a ciscore plot  
source("http://reuningscherer.net/multivariate/r/ciscoreplot.R.txt")  
  
#run the function  
ciscoreplot(pca1, c(1, 2), c(1:dim(pca1$x)[1]))

## Warning in sqrt((5.99 - (y1vec^2)/x$sdev[comps[1]]^2) \* x$sdev[comps[2]]^2):  
## NaNs produced  
## Warning in sqrt((5.99 - (y1vec^2)/x$sdev[comps[1]]^2) \* x$sdev[comps[2]]^2):  
## NaNs produced

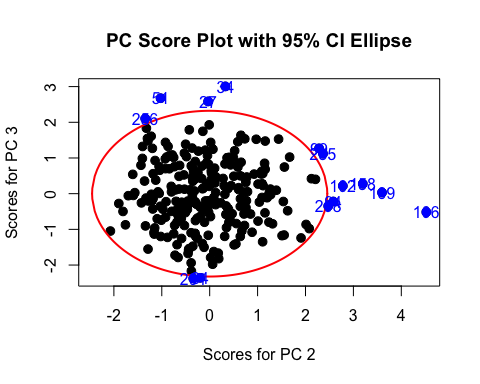


ciscoreplot(pca1, c(1, 3), c(1:dim(pca1$x)[1]))

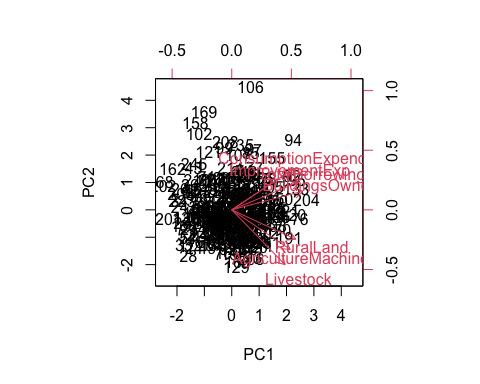
## Warning in sqrt((5.99 - (y1vec^2)/x$sdev[comps[1]]^2) \* x$sdev[comps[2]]^2):  
## NaNs produced  
## Warning in sqrt((5.99 - (y1vec^2)/x$sdev[comps[1]]^2) \* x$sdev[comps[2]]^2):  
## NaNs produced



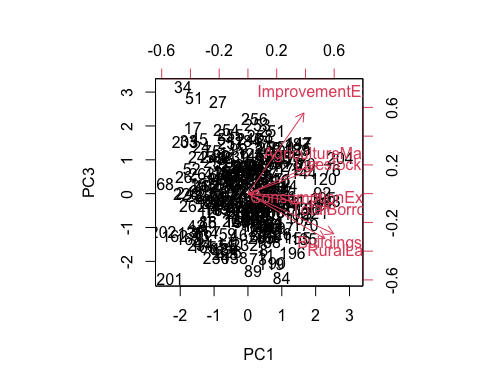
ciscoreplot(pca1, c(2, 3), c(1:dim(pca1$x)[1]))



#make a biplot for first two components  
biplot(pca1, choices = c(1, 2), pc.biplot = T)



biplot(pca1, choices = c(1, 3), pc.biplot = T)



biplot(pca1, choices = c(2, 3), pc.biplot = T)

