Assignment-2

Sai Teja Kanneganti

September 9, 2016

library(asbio) # Contains function ConDis.matrix

## Loading required package: tcltk

library(reshape2) # To access melt function  
library(ggplot2)  
library(VIM)

## Loading required package: colorspace

## Loading required package: grid

## Loading required package: data.table

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:reshape2':  
##   
## dcast, melt

## VIM is ready to use.   
## Since version 4.0.0 the GUI is in its own package VIMGUI.  
##   
## Please use the package to use the new (and old) GUI.

## Suggestions and bug-reports can be submitted at: https://github.com/alexkowa/VIM/issues

##   
## Attaching package: 'VIM'

## The following object is masked from 'package:datasets':  
##   
## sleep

library(Amelia)

## Loading required package: Rcpp

## ##   
## ## Amelia II: Multiple Imputation  
## ## (Version 1.7.4, built: 2015-12-05)  
## ## Copyright (C) 2005-2016 James Honaker, Gary King and Matthew Blackwell  
## ## Refer to http://gking.harvard.edu/amelia/ for more information  
## ##

library(RcppEigen)  
library(HSAUR2)

## Loading required package: tools

library(outliers) # To find outliers in given data.

Question 1

library(asbio) # Contains function ConDis.matrix  
x <- c(3,4,2,1,7,6,5) # Loading value into vector x  
y <- c(4,3,7,6,5,2,1) # Loading vectors into vector y  
ConDis.matrix(x,y) # Function that generates concordant and discordant matrix.

## 1 2 3 4 5 6 7  
## 1 NA NA NA NA NA NA NA  
## 2 -1 NA NA NA NA NA NA  
## 3 -1 -1 NA NA NA NA NA  
## 4 -1 -1 1 NA NA NA NA  
## 5 1 1 -1 -1 NA NA NA  
## 6 -1 -1 -1 -1 1 NA NA  
## 7 -1 -1 -1 -1 1 1 NA

No of concordant pairs are 6 (Number of 1's in lower part of matrix). No of discordant pairs are 15 (Number of -1's in lower part of matrix).

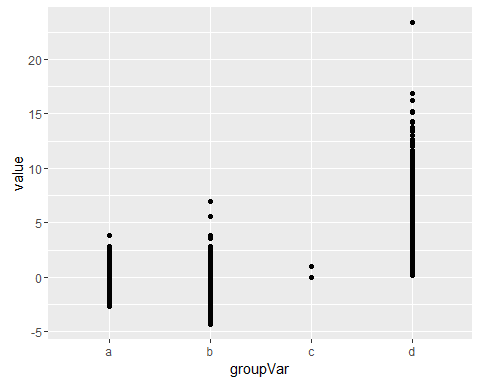
Question 2(a)

library(reshape2) # To access melt function  
a <- rnorm(500) # Generates 500 values of normally distributed data randomly.   
b <- rt(500,5) # Generates 500 values of T distributed data randomly.  
c <- rbinom(500,1,0.5) # Generates 500 values of Binomial distributed data randomly.  
d <- rchisq(500,5) # Generates 500 values of Chi-Squared distributed data randomly.  
df <- data.frame(a,b,c,d) # Dataframe(df) containing 4 variables.  
df2 <- melt(df, variable.name= "groupVar") #Melting all the variables in df to a single variable by name groupVar.

## No id variables; using all as measure variables

Question 2(b)

library(ggplot2) # To access qplot function.  
qplot(groupVar,value,data=df2)

 Question 3(a)

GSAF <- read.csv("E:/Masters/IDA/Assignments/Assignment2/ISE 5103 GSAF.csv", header = TRUE, sep = ",")

Question 3(b)

GSAFdata <- GSAF[c(GSAF$Year>1999), ] #New dataframe containing observations from 2000 year.

Question 3(c)

#Rdate <- as.Date(GSAFdata$Date)  
Rdate <- as.Date(GSAFdata$Date,"%d%b%y")  
#Rdate <- as.Date(GSAFdata$Date, )  
#GSAFdata <- as.factor (GSAFdata)  
GSAFdata <- data.frame(GSAFdata,Rdate)

3(d)

library(VIM) # To access aggr function.  
m <- aggr(GSAFdata$Rdate) # dataframe to find missingness



summary(m)# summary of m

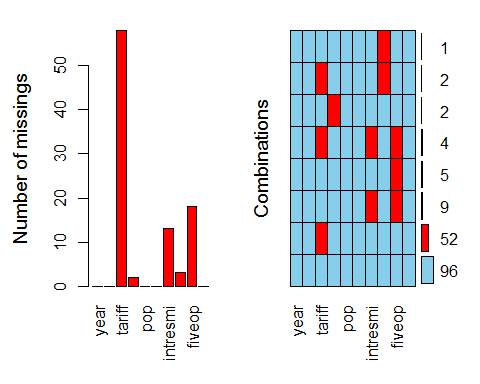
##   
## Missings per variable:   
## Variable Count  
## x 1681  
##   
## Missings in combinations of variables:   
## Combinations Count Percent  
## 1 1681 100

3(e)

GSAFdata <- GSAFdata[!is.na(GSAFdata$Rdate), ]# Deleting null values in dataframe.

Question 4(a):

library(Amelia) # To access freetrade dataframe.  
library(VIM) # To acess aggr function (To explore missingness)  
data(freetrade) # Load data sets.  
miss <- aggr(freetrade,prop= F,numbers= T)# aggr used to calculate and plot missing values.



summary(miss) # Summary of miss

##   
## Missings per variable:   
## Variable Count  
## year 0  
## country 0  
## tariff 58  
## polity 2  
## pop 0  
## gdp.pc 0  
## intresmi 13  
## signed 3  
## fiveop 18  
## usheg 0  
##   
## Missings in combinations of variables:   
## Combinations Count Percent  
## 0:0:0:0:0:0:0:0:0:0 96 56.1403509  
## 0:0:0:0:0:0:0:0:1:0 5 2.9239766  
## 0:0:0:0:0:0:0:1:0:0 1 0.5847953  
## 0:0:0:0:0:0:1:0:1:0 9 5.2631579  
## 0:0:0:1:0:0:0:0:0:0 2 1.1695906  
## 0:0:1:0:0:0:0:0:0:0 52 30.4093567  
## 0:0:1:0:0:0:0:1:0:0 2 1.1695906  
## 0:0:1:0:0:0:1:0:1:0 4 2.3391813

1.From the first plot we can know the number of missing values in each variable of the dataframe. 2.From second graph(combinations) we can get number of missing values of each variable by adding numbers corresponding to the red squares(which represent missing values) for a given variable. eg: Number of missing values in tariff=(52+4+2 = 58) 3.Summary gives the number of missing values in each variable and the missings in combination of variables.

Question 4(b)

tab <- table(freetrade$country,freetrade$tariff) # creating table for country and tariff variables.  
chisq.test(tab) # To perform Chi-squared test for tab.

## Warning in chisq.test(tab): Chi-squared approximation may be incorrect

##   
## Pearson's Chi-squared test  
##   
## data: tab  
## X-squared = 831.96, df = 736, p-value = 0.007819

freetrade2 <- freetrade[!(freetrade$country =='Nepal'), ] #creating dataframe removing observations(rows) where Nepal is the country.  
tab2 <- table(freetrade2$country,freetrade2$tariff)   
chisq.test(tab2)

## Warning in chisq.test(tab2): Chi-squared approximation may be incorrect

##   
## Pearson's Chi-squared test  
##   
## data: tab2  
## X-squared = 684.79, df = 602, p-value = 0.01063

freetrade3 <- freetrade[!(freetrade$country =='Philippines'), ] ##creating dataframe removing observations(rows) where philippines is the country.  
tab3 <- table(freetrade3$country,freetrade3$tariff)  
chisq.test(tab3) # p < 0.05

## Warning in chisq.test(tab3): Chi-squared approximation may be incorrect

##   
## Pearson's Chi-squared test  
##   
## data: tab3  
## X-squared = 639.33, df = 574, p-value = 0.03012

In all the above three observations p < 0.05 (Reject null hypothesis,Not independent).So,missingness in the tariff variable is dependent with country variable.

No,the answer dosenot change if we remove Nepal or Philippines

problem 5(a) - 1

corMat <- cor(mtcars, use="complete.obs", method="kendall") # creating correlation matrix by kendall method.

Problem 5(a) - 2

library(RcppEigen) # Library to access eigen  
eigen(corMat) # Gives Eigen values and Eigen vectors.

## $values  
## [1] 5.9617011 2.1893862 0.6984422 0.4642294 0.4185584 0.3169916 0.2578143  
## [8] 0.2361555 0.2114008 0.1532397 0.0920809  
##   
## $vectors  
## [,1] [,2] [,3] [,4] [,5]  
## [1,] 0.3575297 0.02675153 0.27314200 -0.122875225 0.22839328  
## [2,] -0.3836368 -0.04418909 0.11327763 0.141687777 -0.14643170  
## [3,] -0.3575397 0.02766019 -0.12944823 0.230388848 -0.27522963  
## [4,] -0.3367405 -0.18588965 -0.18000166 0.002993274 -0.19168715  
## [5,] 0.2618797 -0.28571501 -0.35388191 0.790456353 0.25045750  
## [6,] -0.3422897 0.14434532 -0.27899868 0.028116399 -0.09643668  
## [7,] 0.1743478 0.48434024 -0.48940055 -0.177540345 -0.17229755  
## [8,] 0.3126649 0.27554274 -0.39234525 0.005629932 -0.10474969  
## [9,] 0.2475438 -0.44361595 0.09684932 -0.043579408 -0.52622105  
## [10,] 0.2402172 -0.43984706 -0.30037584 -0.286694541 -0.33352732  
## [11,] -0.2255762 -0.39846352 -0.40899680 -0.412893811 0.55951939  
## [,6] [,7] [,8] [,9] [,10]  
## [1,] 0.14481402 -0.11648693 0.17634648 -0.795505357 -0.14449930  
## [2,] -0.02763314 0.18339135 -0.01798676 -0.198283980 0.34344276  
## [3,] 0.39217141 -0.11048019 -0.56193172 -0.309163387 -0.35748934  
## [4,] -0.71460574 0.03321574 0.17359735 -0.393683969 -0.07557976  
## [5,] 0.03864551 0.12636076 0.09707612 -0.068000660 0.07365039  
## [6,] 0.37899549 -0.36867309 0.70256342 0.005013438 -0.02661174  
## [7,] 0.09906505 0.62483268 0.05158263 -0.169063949 0.04822787  
## [8,] -0.33716562 -0.51735600 -0.16550166 0.061404155 -0.24314084  
## [9,] 0.08304584 0.24428095 0.24660367 0.138874346 -0.51464989  
## [10,] 0.17795425 -0.23734563 -0.14611316 -0.143249852 0.56988656  
## [11,] 0.08133639 0.11628895 -0.08626688 0.021833583 -0.26541465  
## [,11]  
## [1,] 0.101199464  
## [2,] 0.777516362  
## [3,] -0.143739020  
## [4,] -0.283367778  
## [5,] -0.008401892  
## [6,] -0.013610561  
## [7,] -0.022324169  
## [8,] 0.435615970  
## [9,] 0.205440097  
## [10,] -0.102613133  
## [11,] 0.202529509

problem 5(a) - 3

pc <- prcomp (corMat,scale= T) # principal compnents of correlation matrix are obtained by centering and scaling by a factor 2.

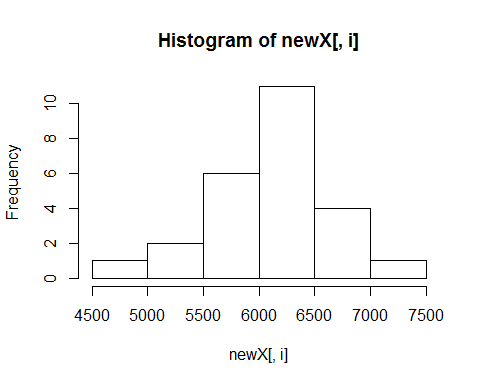
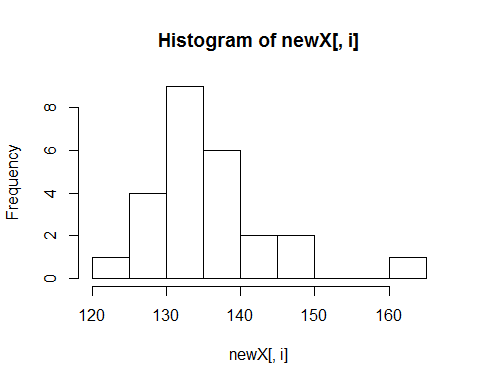
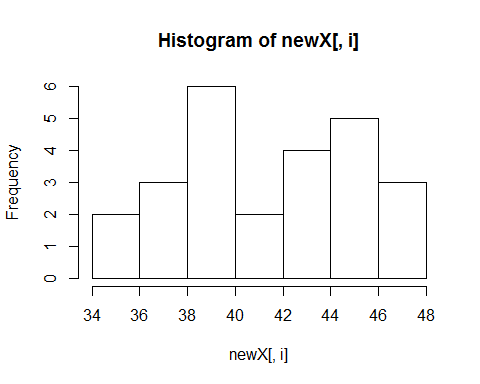
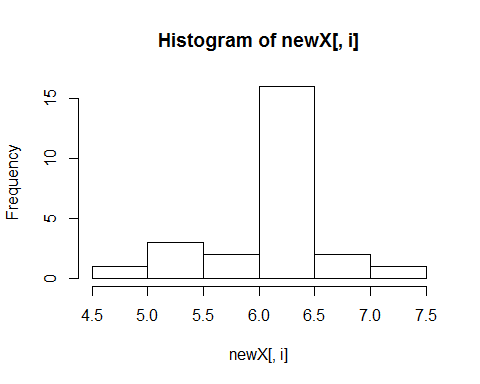
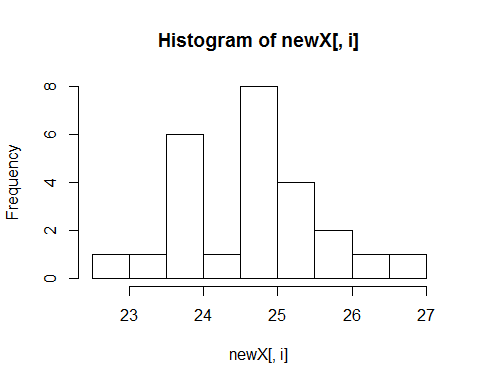
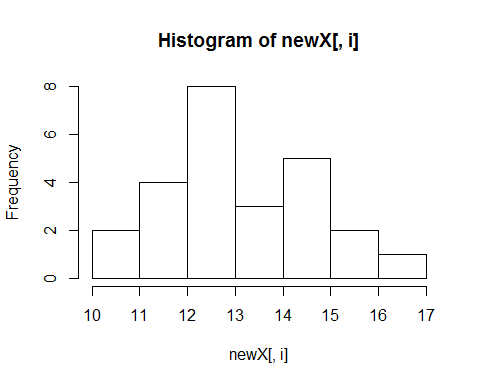
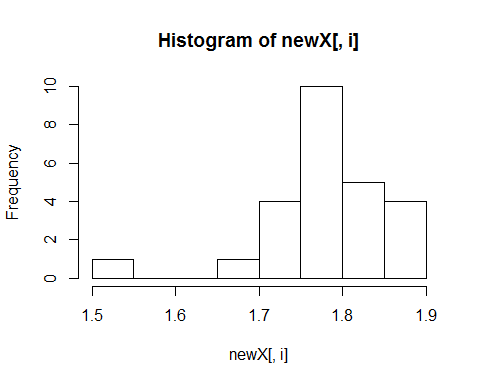
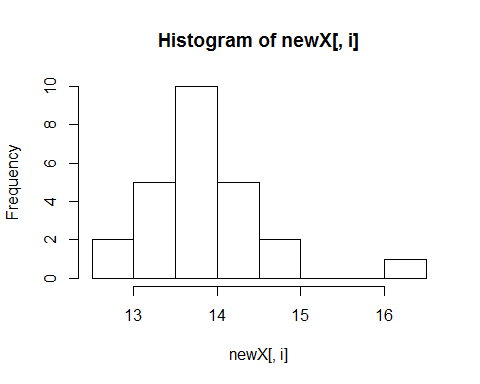
problem 5(a) - 4 Results are different since principal componets are diffrent from eigen values. problem 5(a) - 5

angle <- function(a,b){  
 inner.prod <- a%\*%b   
 xlength <- norm(a,type="2")# type 2 for row sum  
 ylength <- norm(b,type="2")  
 theta <- acos(inner.prod / (xlength \* ylength))  
 as.numeric(theta) # convert from matrix to numeric.  
}  
angle(pc$x[,1],pc$x[,2])

## [1] 1.570796

problem 5(b) - 1

library(HSAUR2) # To access heptathlon  
data("heptathlon") # loads heptathlon  
apply(heptathlon[,1:8],2,hist)



## $hurdles  
## $breaks  
## [1] 12.5 13.0 13.5 14.0 14.5 15.0 15.5 16.0 16.5  
##   
## $counts  
## [1] 2 5 10 5 2 0 0 1  
##   
## $density  
## [1] 0.16 0.40 0.80 0.40 0.16 0.00 0.00 0.08  
##   
## $mids  
## [1] 12.75 13.25 13.75 14.25 14.75 15.25 15.75 16.25  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $highjump  
## $breaks  
## [1] 1.50 1.55 1.60 1.65 1.70 1.75 1.80 1.85 1.90  
##   
## $counts  
## [1] 1 0 0 1 4 10 5 4  
##   
## $density  
## [1] 0.8 0.0 0.0 0.8 3.2 8.0 4.0 3.2  
##   
## $mids  
## [1] 1.525 1.575 1.625 1.675 1.725 1.775 1.825 1.875  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $shot  
## $breaks  
## [1] 10 11 12 13 14 15 16 17  
##   
## $counts  
## [1] 2 4 8 3 5 2 1  
##   
## $density  
## [1] 0.08 0.16 0.32 0.12 0.20 0.08 0.04  
##   
## $mids  
## [1] 10.5 11.5 12.5 13.5 14.5 15.5 16.5  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $run200m  
## $breaks  
## [1] 22.5 23.0 23.5 24.0 24.5 25.0 25.5 26.0 26.5 27.0  
##   
## $counts  
## [1] 1 1 6 1 8 4 2 1 1  
##   
## $density  
## [1] 0.08 0.08 0.48 0.08 0.64 0.32 0.16 0.08 0.08  
##   
## $mids  
## [1] 22.75 23.25 23.75 24.25 24.75 25.25 25.75 26.25 26.75  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $longjump  
## $breaks  
## [1] 4.5 5.0 5.5 6.0 6.5 7.0 7.5  
##   
## $counts  
## [1] 1 3 2 16 2 1  
##   
## $density  
## [1] 0.08 0.24 0.16 1.28 0.16 0.08  
##   
## $mids  
## [1] 4.75 5.25 5.75 6.25 6.75 7.25  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $javelin  
## $breaks  
## [1] 34 36 38 40 42 44 46 48  
##   
## $counts  
## [1] 2 3 6 2 4 5 3  
##   
## $density  
## [1] 0.04 0.06 0.12 0.04 0.08 0.10 0.06  
##   
## $mids  
## [1] 35 37 39 41 43 45 47  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $run800m  
## $breaks  
## [1] 120 125 130 135 140 145 150 155 160 165  
##   
## $counts  
## [1] 1 4 9 6 2 2 0 0 1  
##   
## $density  
## [1] 0.008 0.032 0.072 0.048 0.016 0.016 0.000 0.000 0.008  
##   
## $mids  
## [1] 122.5 127.5 132.5 137.5 142.5 147.5 152.5 157.5 162.5  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"  
##   
## $score  
## $breaks  
## [1] 4500 5000 5500 6000 6500 7000 7500  
##   
## $counts  
## [1] 1 2 6 11 4 1  
##   
## $density  
## [1] 0.00008 0.00016 0.00048 0.00088 0.00032 0.00008  
##   
## $mids  
## [1] 4750 5250 5750 6250 6750 7250  
##   
## $xname  
## [1] "newX[, i]"  
##   
## $equidist  
## [1] TRUE  
##   
## attr(,"class")  
## [1] "histogram"

problem 5(b) - 2

apply(heptathlon[,1:8],2,grubbs.test)

## $hurdles  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Launa (PNG) = 3.5024, U = 0.4676, p-value = 0.000436  
## alternative hypothesis: highest value 16.42 is an outlier  
##   
##   
## $highjump  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Launa (PNG) = 3.61810, U = 0.43184, p-value = 0.0001698  
## alternative hypothesis: lowest value 1.5 is an outlier  
##   
##   
## $shot  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Hui-Ing (TAI) = 2.08970, U = 0.81047, p-value = 0.3702  
## alternative hypothesis: lowest value 10 is an outlier  
##   
##   
## $run200m  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Joyner-Kersee (USA) = 2.15480, U = 0.79847, p-value = 0.3048  
## alternative hypothesis: lowest value 22.56 is an outlier  
##   
##   
## $longjump  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Launa (PNG) = 2.68320, U = 0.68752, p-value = 0.04594  
## alternative hypothesis: lowest value 4.88 is an outlier  
##   
##   
## $javelin  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Scheider (SWI) = 1.69720, U = 0.87498, p-value = 1  
## alternative hypothesis: highest value 47.5 is an outlier  
##   
##   
## $run800m  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Launa (PNG) = 3.30190, U = 0.52681, p-value = 0.001808  
## alternative hypothesis: highest value 163.43 is an outlier  
##   
##   
## $score  
##   
## Grubbs test for one outlier  
##   
## data: newX[, i]  
## G.Launa (PNG) = 2.68190, U = 0.68781, p-value = 0.04618  
## alternative hypothesis: lowest value 4566 is an outlier

match\_vector = NULL  
index = NULL  
outlier\_identification = function(){  
 for (d in 1:ncol(heptathlon)) {  
   
 match\_vector[d] <- match(outlier(heptathlon[, d]), heptathlon[,d])   
  
 }  
 return(match\_vector)  
}  
index <- names(which.max(table(outlier\_identification()))) #returns the name of index of first max in a table.  
heptathlon <- heptathlon[-as.numeric(index),]

problem 5(b) - 3

largeValues <- function(y,u){  
 for (v in 1:nrow(heptathlon)) {  
 heptathlon[v,u] <- y - heptathlon[v,u]  
   
 }  
   
 return(heptathlon)  
}  
heptathlon <- largeValues(max(heptathlon$run200m), "run200m")  
heptathlon <- largeValues(max(heptathlon$run800m), "run800m")  
heptathlon <- largeValues(max(heptathlon$hurdles), "hurdles")  
  
head(heptathlon)

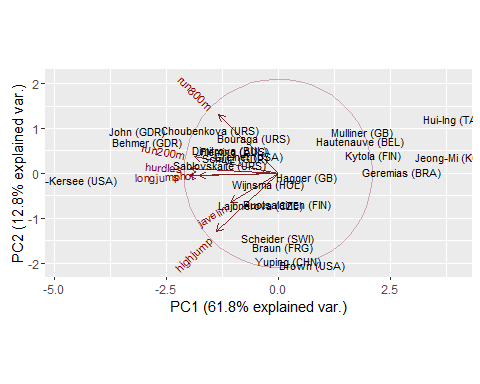
## hurdles highjump shot run200m longjump javelin  
## Joyner-Kersee (USA) 2.16 1.86 15.80 4.05 7.27 45.66  
## John (GDR) 2.00 1.80 16.23 2.96 6.71 42.56  
## Behmer (GDR) 1.65 1.83 14.20 3.51 6.68 44.54  
## Sablovskaite (URS) 1.24 1.80 15.23 2.69 6.25 42.78  
## Choubenkova (URS) 1.34 1.74 14.76 2.68 6.32 47.46  
## Schulz (GDR) 1.10 1.83 13.50 1.96 6.33 42.82  
## run800m score  
## Joyner-Kersee (USA) 18.16 7291  
## John (GDR) 20.55 6897  
## Behmer (GDR) 22.47 6858  
## Sablovskaite (URS) 14.43 6540  
## Choubenkova (URS) 18.77 6540  
## Schulz (GDR) 20.88 6411

problem 5(b) - 4

heptathlon[,8] <- NULL  
Hpca <- prcomp(heptathlon, scale. = T)

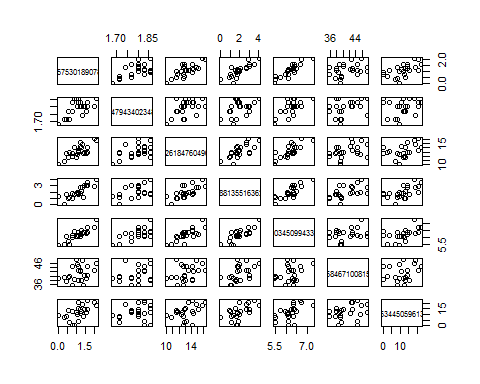
problem 5(b) - 5

ggbiplot::ggbiplot(Hpca, obs.scale = 1, var.scale = 1, ellipse = TRUE, circle=TRUE, labels = row.names(heptathlon), labels.size = 3)

 1.Vector represent event and dot represent names 2.shorts, longjumps, and hurdles are in correlation with PC1. 3.run200m and high jump are in correlation with PC2

problem 5(b) - 6

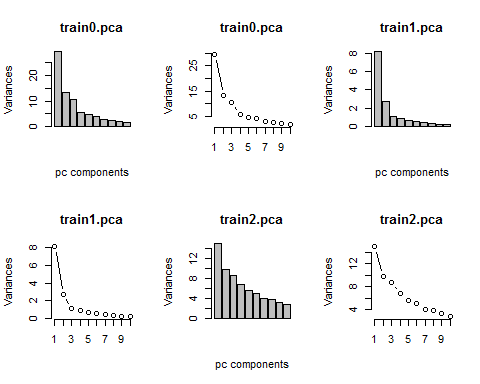
plot(heptathlon, Hpca$x[,1])

 problem 5(c) - 1

train0 <- read.csv("train.0")  
train1 <- read.csv("train.1")  
train2 <- read.csv("train.2")

Principal components are used in images to analyse and compress the image without losing much data. problem 5(c) - 2

par(mfrow = c(2,3))  
train0.pca <- prcomp(train0) #training data set 1  
plot(train0.pca, xlab = "pc components")# plots the proportion of variance.  
screeplot(train0.pca, type = "line")  
train1.pca <- prcomp(train1) # for training data set 2  
plot(train1.pca, xlab = "pc components")  
screeplot(train1.pca, type = "line") # to describe major percentage of data better to approx 7 Principal components  
train2.pca <- prcomp(train2) # for training data set 3  
plot(train2.pca, xlab = "pc components")  
screeplot(train2.pca, type = "line")



train2.summary <- summary(train2.pca)

problem 6(a)

crime <- read.csv("E:/Masters/IDA/Assignments/Assignment2/test.csv", header = TRUE, sep = ",") # Reading downloaded file and named dataframe as crime.

Url <https://www.kaggle.com/account/login?ReturnUrl=%2fc%2fsf-crime%2fdownload%2ftest.csv.zip>

Description of data variables:

Dates - Date on which crime took place. Day Of Week - Day of the week on which crime occurred. PdDistrict - name of the police department district. Address - Address of crime incident X - Longitude Y - Latitude

This dataframe gives information about crimes that took place in San Fransisco from 2003 to 2015.

problem 6(b)

ncol(crime) # To find number of variables (Columns) in crime dataframe.

## [1] 7

nrow(crime) # To find number of rows (Observations) in crime dataframe.

## [1] 884262

mean(crime$X) # Mean value of Longitude in dataframe

## [1] -122.4227

median(crime$X) # Median value of Longitude in dataframe

## [1] -122.4165

sd(crime$X) # Standard deviation of Longitude in dataframe

## [1] 0.03098498

var(crime$X) # variance of Longitude in dataframe

## [1] 0.0009600688

mad(crime$X) # Median Absolute Deviation of Longitude in dataframe

## [1] 0.01738766

mean(crime$Y) # Mean value of Latitude in dataframe

## [1] 37.77148

median(crime$Y) # Median value of Latitude in dataframe

## [1] 37.77542

sd(crime$Y) # Standard deviation of Latitide in dataframe

## [1] 0.4848236

var(crime$Y) # variance of Latitude in dataframe

## [1] 0.2350539

mad(crime$Y) # Median Absolute Deviation of Latitude in dataframe

## [1] 0.01822512

library(outliers) # To find outliers in given data.   
outlier(crime$X) # Finding outliers in Longitude.

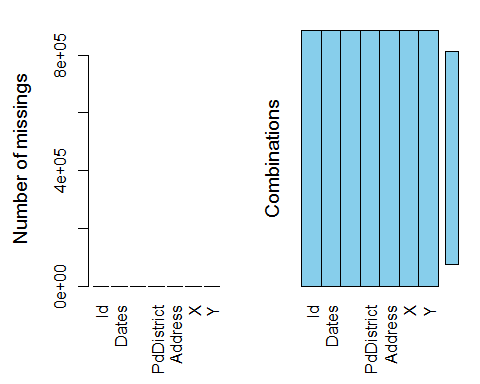
## [1] -120.5

outlier(crime$Y) # Finding outliers in Latitude.

## [1] 90

library(VIM) # Library to find missing values.  
miss\_crime <- aggr(crime,prop= F,numbers= T) #

## Warning in plot.aggr(res, ...): not enough horizontal space to display  
## frequencies



summary(miss\_crime) # Gives summary of miss\_crime

##   
## Missings per variable:   
## Variable Count  
## Id 0  
## Dates 0  
## DayOfWeek 0  
## PdDistrict 0  
## Address 0  
## X 0  
## Y 0  
##   
## Missings in combinations of variables:   
## Combinations Count Percent  
## 0:0:0:0:0:0:0 884262 100

1. There is 1 outlier in X,i.e. -120.5
2. There is 1 outlier in Y,i.e. 90
3. There are no missing values in the given data. This can be observed from the plots from aggr function and by doing summary of the dataframe.